BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.

대 한 민 국 특 허 청
KOREAN INTELLECTUAL PROPERTY OFFICE

별첨 사본은 아래 출원의 원본과 동일함을 증명함.

This is to certify that the following application annexed hereto is a true copy from the records of the Korean Intellectual Property Office.

출 원 번 호 : 10-2002-0066269
Application Number

출 원 년 월 일 : 2002년 10월 29일
Date of Application OCT 29, 2002

출 원 인 : 주식회사 하이닉스반도체
Applicant(s) Hynix Semiconductor Inc.

2003 년 05 월 14 일

특 허 청
COMMISSIONER

【서지사항】

【서류명】 특허출원서
【권리구분】 특허
【수신처】 특허청장
【제출일자】 2002.10.29
【발명의 명칭】 고속 데이터 억세스를 위한 반도체 메모리 장치
【발명의 영문명칭】 Semiconductor Memory device for high speed Data access
【출원인】
【명칭】 주식회사 하이닉스반도체
【출원인코드】 1-1998-004569-8
【대리인】
【명칭】 특허법인 신성
【대리인코드】 9-2000-100004-8
【지정된변리사】 변리사 정지원, 변리사 원석희, 변리사 박해천
【포괄위임등록번호】 2000-049307-2
【발명자】
【성명의 국문표기】 안진홍
【성명의 영문표기】 AHN,Jin Hong
【주민등록번호】 581124-1110419
【우편번호】 431-070
【주소】 경기도 안양시 동안구 평촌동 130-1 영풍아파트 101-1408
【국적】 KR
【발명자】
【성명의 국문표기】 홍상훈
【성명의 영문표기】 HONG,Sang Hoon
【주민등록번호】 700930-1064113
【우편번호】 467-860
【주소】 경기도 이천시 부발읍 신하리 청구 아파트 101-1302
【국적】 KR
【발명자】
【성명의 국문표기】 김세준
【성명의 영문표기】 KIM,Se Jun
【주민등록번호】 740209-1069419

【우편번호】 463-500

【주소】 경기도 성남시 분당구 구미동 까치마을 선경아파트 107-1002

【국적】 KR

【발명자】

【성명의 국문표기】 고재범

【성명의 영문표기】 KO,Jae Bum

【주민등록번호】 760926-1026025

【우편번호】 131-811

【주소】 서울특별시 중랑구 면목8동 2-8 3층 6/6

【국적】 KR

【취지】 특허법 제42조의 규정에 의하여 위와 같이 출원합니다. 대리인 특허법인 신성 (인)

【수수료】

【기본출원료】 20 면 29,000 원

【가산출원료】 55 면 55,000 원

【우선권주장료】 0 건 0 원

【심사청구료】 0 항 0 원

【합계】 84,000 원

【첨부서류】 1. 요약서·명세서(도면)_1통

【요약서】

【요약】

데이터 억세스 패턴에 상관없이 고속으로 데이터를 억세스 할 수 있는 메모리 장치을 제공기 위한 것으로, 이를 위해 본 발명은 다수개의 단위셀을 구비하는 제1 셀블럭; 상기 제1 셀블럭에 억세스되는 데이터 재저장을 위한 제2 셀블럭; 및 상기 제1 셀블럭에 대해 연속적으로 제1 데이터 및 제2 데이터가 억세스될 때, 상기 제1 셀블럭에서는 상기 제1 데이터의 재저장동작을 수행하지 않고 상기 제2 데이터가 억세스되도록 제어하고, 상기 제2 셀블럭에서는 상기 제1 데이터의 재저장 동작이 수행하도록 제어하기 위한 제어부를 구비하는 메모리 장치를 제공한다.

【대표도】

도 5a

【색인어】

반도체, 메모리, 태그메모리, 센스앰프, 예비블럭. 재저장, 고속 억세스.

【명세서】

【발명의 명칭】

고속 데이터 억세스를 위한 반도체 메모리 장치{Semiconductor Memory device for high speed Data access}

【도면의 간단한 설명】

도1은 종래기술에 의한 메모리 장치의 블럭구성도.

도2는 도1에 도시된 비트라인 센스앰프부 및 셀블럭을 나타내는 회로도.

도3는 도1에 도시된 메모리 장치의 동작을 보여주는 파형도.

도4는 통상적인 메모리 장치에서 뱅크간 인터리빙 모드를 사용할 때의 동작을 보여주는 파형도.

도5a는 본 발명의 개념을 나타내기 위한 메모리 장치의 블럭구성도.

도5b는 본 발명의 개념을 나타내기 위한 메모리 장치의 블럭구성도.

도5c는 본 발명의 바람직한 제1 실시예에 따른 메모리 장치의 블럭구성도.

도6은 도5c의 도시된 셀블럭 및 센스앰프부를 나타내는 회로도.

도7은 도5c에 도시된 셀블럭 영역의 일례를 나타내는 블럭구성도.

도8a 및 도8d은 도6에 도시된 셀블럭간의 데이터 이동을 보여주는 블럭구성도.

도9 내지 도11은 본 발명에 의한 메모리 장치에서 동작을 나타내는 파형도.

도12은 도15에 도시된 순서대로 리드명령어가 입력될 때 도5에 도시된 메모리 장치의 시뮬레이션 파형도.

도13은 도5c에 도시된 태그제어부를 나타내는 블럭구성도.

도14는 도5c에 도시된 예비블럭 테이블을 나타내는 블럭구성도.

도15은 도5c에 도시된 태그테이블을 나타내는 블럭구성도.

도16은 도15에 도시된 태그테이블의 일예를 나타내는 회로도.

도17는 도14의 예비블럭테이블의 일예를 나타내는 회로도.

도18은 본 발명에 의한 메모리 장치에서 태그테이블 및 예비블럭 테이블의 동작을 나타내는 파형도.

도19은 도6에 도시된 글로벌 비트라인 연결부를 제어하기 위한 제어부의 일례를 나타내는 회로도.

도20는 도19에 도시된 제어부에서 출력되는 제어신호와 셀블럭 내부의 동작을 나타내는 파형도.

도21a 및 도22b는 본 발명의 제2 실시예에 따른 메모리 장치의 블럭구성도.

도22는 도21b에 도시된 메모리 장치의 동작을 나타내는 블럭구성도.

【발명의 상세한 설명】

【발명의 목적】

【발명이 속하는 기술분야 및 그 분야의 종래기술】

<23> 본 발명은 반도체 메모리 장치에 관한 것으로, 특히 데이터를 고속으로 억세스(Access)할 수 있는 메모리 장치에 관한 것이다.

<24> 반도체 메모리 장치는 크게 RAM(Random Access Memory)과 ROM(Read only Memory)으로 구분할 수가 있다.

<25> 램(RAM)은 1개의 트랜지스터(transister)와 1개의 캐패시터가 하나의 단위셀(unit cell)을 구성하는 다이나믹 램(Dynimic RAM)과, 6개의 트랜지스터 또는 4개의 트랜지스터 및 2개의 부하 저항으로 구성되는 스태틱램(Static RAM)으로 나뉘어 지는데, 집적도면에서나 제조공정등에서 효율적인 다이나믹 램이 컴퓨터의 메인 메모리등 여러분야에 널리 사용되고 있다.

<26> 근래에 중앙처리장치(CPU)의 동작속도는 메모리 장치(DRAM)의 동작속도를 능가할 정도로 현저히 향상되어 왔으며, 그 결과 메모리 장치의 동작속도가 중앙처리장치의 동작속도보다 상대적으로 느려 여러가지 문제점이 발생하고 있다. 이러한 문제점을 극복하기 위해 보다 고속으로 데이터를 입출력하기 위한 다양한 구조의 이 개발되고 있다.

<27> 도1은 종래기술에 의한 메모리 장치의 대략적인 블럭구성도이다.

<28> 도1을 참조하여 살펴보면, 메모리 장치은 외부에서 입력되는 다수의 명령어신호(/RAS,/CAS,/WE,/CS,CKE,CK 등)를 입력받아 뱅크(100)의 동작(예컨대 리드(read), 라이트(write), 리프레쉬(refresh) 동작)을 제어하기 위한 명령어제어부(200)와, 독립적으로 로우디코더(row decoder)와 칼럼디코더(column decoder)를 구비하여, 입력되는 어드레스에 대응하는 단위셀의 데이터를 리드(read)하거나, 상기 단위셀에 데이터를 라이트(write)하는 뱅크(100)와, 뱅크(100)로 부터 입,출력되는 데이터를 버퍼링하여 외부로 입출력하기 위한 데이터 입,출력버퍼(300)를 구비한다.

<29> 통상 메모리 장치은 다수의 뱅크(예컨대 4개의 뱅크)를 구비하고 있는데, 각각의 뱅크는 같은 구조를 가지고 있기 때문에, 도1에는 하나의 뱅크(100)만 도시하였다. 또한 도1에서는 메모리 장치에 구비되는 블럭중에서 본 발명을 설명하기 위해 필요한 최소한의 블럭만을 도시한 것이다.

<30> 하나의 뱅크(100)에는 총 8개의 세그먼트(120a ~ 120h)를 구비하고, 세그먼트(120a ~ 120d)에서 출력되는 데이터를 증폭하여 데이터 입,출력버퍼(300)로 전달하거나, 데이터입출력버퍼(300)에서 입력되는 데이터를 세그먼트로 전달하기 위한 I/O 센스앰프부(110,130)를 구비하고 있다.

<31> 하나의 세그먼트(예컨대 120a)는 로우어드레스를 디코딩(decoding)하여 셀영역(123a~123i,124a~124h)으로 출력하는 로우어드레스 디코더부(121)와, 컬럼어드레스를 디코딩하여 셀영역(123a~123i,124a~124h)으로 출력하는 컬럼어드레스 디코더부(122)와, 다수개의 단위셀로 구성된 다수의 셀블럭(Cell block)(124a~124h)과, 셀블럭에서 출력되는 데이터를 감지 증폭하기 위해 셀블럭(124a~124h)간에 배치된 비트라인 센스앰프부(123a~123i)를 구비하고 있다.

<32> 도1에 도시된 메모리 장치은 용량이 256Mb인 경우를 나타내고 있는데, 4개의 뱅크로 구성되는 경우 하나의 뱅크에는 64Mb의 단위셀이 구비되며, 하나의 세그먼트는 8Mb로 총 8개의 세그먼트가 하나의 뱅크를 구성한다. 하나의 세그먼트에는 총 8개의 셀블럭(124a~124h)을 구비하고 있고, 하나의 셀블럭에는 256개의 워드라인(Word Line)과 4K(4 * 1024)개의 비트라인이 구비된다. 따라서 하나의 셀블럭에는 245 * 4K개의 단위셀이 구비된다. 이하에서는 전술한 바와 같이 하나의 세그먼트가 8Mb이고, 8개의 셀블럭이 각각 256개의 워드라인을 구비한 경우에 대해 설명한다.

<33> 또한 8개의 셀블럭(124a~124h) 사이에 9개의 비트라인 센스앰프부(123a~123i)가 구비되어, 하나의 센스앰프부(예컨대 123b)는 이웃한 두 셀블럭(124a,124b)에 공유되도록 구성되어 있다. 셀블럭의 회로구조상 하나의 셀블럭에는 일측과 타측에 각각 2개의 센스앰프부가 필요하기 때문에 8개의 셀블럭에는 총16개의 비트라인 센스앰프부가 필요하지만, 회로면적등의 효율을 위해서 하나의 비트라인 센스앰프부를 2개의 이웃한 셀블럭이 공유하고, 셀블럭과 비트라인 센스앰프간에 연결부를 두고, 적절한 타이밍에 이웃한 2개의 셀블럭과 비트라인 센스앰프부를 연결 또는 분리하도록 하고 있다..

<34> 도2는 도1에 도시된 비트라인 센스앰프부 및 셀블럭의 일예를 나타내는 회로도로서, 특히 셀블럭0 및 셀블럭1(124a,124b)과 비트라인 센스앰프(123b)의 일부분을 도시하고 있다.

<35> 도2를 참조하여 살펴보면, 셀블럭0(124a)에는 하나의 모스트랜지스터와 하나의 캐패시터로 구성된 단위셀이 256 * 4k개 구비되어 있고, 워드라인(WL)이 각 단위셀을 구성하는 모스트랜지스터의 게이트단으로 연결되고, 비트라인(BL, /BL)이 워드라인과 교차하면서 단위셀을 구성하는 모스트랜지스터의 드레인단으로 연결되어 있다. 도시되지 않았지만 셀블럭1(124b)에서도 셀블럭0(124a)와 같은 구성이다.

<36> 비트라인 센스앰프부(123b)는 비트라인 센스앰프 인에이블신호(RTO, /S)에 의해 인에이블되어, 비트라인(예컨대 BL0,/BL0)의 신호 차이를 증폭하기 위한 센스앰프(123b_4)와, 센스앰프가 디스에이블일 때에 출력되는 프리차지 인에이블신호(BLEQ')에 인에이블되어 비트라인 프리차지 전압(Vblp)으로 비트라인(예컨대 BL0, /BL0)을 프리차지하기 위한 프리차지부(123b_3)와, 이퀄라이제이션 신호(BLEQ)에 의해 셀블럭0(124a)에 연결된 비트라인(예컨대 BL0,/BL0)의 전압레벨을 같게 하기 위한 이퀄라이제이션부(123b_2)와,

칼럼어드레스에 의해 생성되는 컬럼제어신호(예컨대 CD0)에 의해 비트라인 센스앰프(123b_4)에 의해 증폭된 데이터신호를 데이터 라인(DB0, /DB0)으로 출력하기 위한 데이터출력부(123b_5)와, 셀블럭(124a,124b)과 비트라인 센스앰프부(123b)와의 연결을 위한 연결부(123b_1,123b_6)를 구비한다.

<37> 여기서 비트라인 센스앰프부(123b)에 구비되는 센스앰프의 수는 셀블럭에 구비되는 비트라인의 수에 따라 정해지고, 센스앰프 연결신호(BISH,BISL)에 의해서 셀블럭0(124a) 또는 셀블럭1(124b)과 비트라인센스앰프부(123b)가 연결된다.

<38> 도3 및 도4는 도1에 도시된 메모리 장치의 동작을 나타내는 파형도이다.

<39> 이하에서는 도1 내지 도4를 참조하여, 종래 기술에 의한 메모리 장치의 동작을 살펴본다.

<40> 먼저, 메모리 장치에 입력되는 어드레스를 살펴보면, 어드레스는 크게 셀블럭어드레스와 로우어드레스, 컬럼어드레스로 구분된다. 셀블럭어드레스는 뱅크 및 세그먼트와 세그먼트내의 셀블럭을 선택하기 위한 어드레스 신호이고, 로우어드레스는 셀블럭내에서 하나의 워드라인을 선택하기 위한 어드레스 신호이며, 컬럼어드레스는 하나의 워드라인에 의해 선택된 4K개의 단위셀을 선택하기 위한 신호이다.

<41> 이어서 메모리 장치의 동작중일 때, 모든 셀블럭의 비트라인(BL,/BL)이 도2에 도시된 프리차지부(123b_3)에 의해 프리차지(통상적으로 전원전압의 1/2)되어 있는 상태에서 리드명령어와 어드레스가 입력되면, 입력된 어드레스에 해당되는 뱅크(100)의 세그먼트(예컨대 120a)가 선택이 된다.

<42> 이어서 로우어드레스 디코더부(121)에서 입력된 로우어드레스를 디코딩하여 세그먼트(120a)에 구비된 8개의 셀블럭중에서 하나의 블럭(예컨대 셀블럭0(124a)을 선택하고, 선택된 블럭에 구비된 256개의 워드라인중 하나의 워드라인(예컨대 셀블럭0의 WL0)을 활성화시킨다. 이 때 센스앰프 연결신호(BISH)가 인에이블되고, 센스앰프 연결신호(BISL)은 디스에이블상태가 되어 비트라인센스앰프부(123b)는 셀블럭0(124a)와 연결된 상태이다.

<43> 이어서 활성화된 워드라인(WL0)에 연결된 4K개의 단위셀에 저장된 데이터가 각각의 비트 라인(BL0,BL1,...)에 인가된다. 이어서 비트라인 센스앰프(123b_4)는 비트라인에 인가된 데이터신호를 감지, 증폭한다.

<44> 여기서 비트라인센스앰프부(123b)는 셀블럭0(124a)의 일측에 구비되어 짝수번째의 비트라인(BL0,/BL0,BL2,/BL2,..,)에 인가된 데이터신호를 감지 증폭하고, 홀수번째의 비트라인(BL1,/BL1,BL3,/BL3,..)에 인가된 데이터신호를 감지 증폭하기 위한 비트라인센스앰프부(미도시)는 셀블럭0(124a)의 타측에 구비된다. 이렇게 4개의 비트라인당 하나의 센스앰프가 구비되어 이웃한 셀블럭과 적절한 타이밍에 연결또는 분리하도록 하는 이유는 전술한 바와 같이 메모리 장치의 고집적을 위한 것이다.

<45> 이어서, 컬럼어드레스 디코더부(122)는 입력된 컬럼어드레스를 디코딩하여 비트라인 센스앰프부의 컬럼선택신호(CD0,CD1,..)(도2의 123b_5 참조)를 출력하고, 비트라인센스앰프(123b_4)에 의해 증폭이 끝난 4K의 데이터중에서 컬럼선택신호에 의해 선택된 데이터가 데이터라인(DB,/DB)를 통해 I/O 센스앰프부(도1의 110)로 출력된다. 이어서 I/O 센스앰프(110)는 비교적 긴 데이터 라인으로 인해 줄어들 데이터신호를 한번 더 증폭하

여 데이터 입,출력버퍼(300)로 전달하고, 데이터 입,출력버퍼(300)에서는 각 뱅크에서 출력되는 데이터를 프리패치하여 데이터가 출력될 타이밍에 외부로 출력한다.

<46> 한편, 비트라인 센스앰프에 의해 증폭된 데이터가 외부로 출력되는 타이밍에 비트라인 센스앰프부(123b_)에서는 활성화되었던 워드라인(WL0)에 대응되는 4K개의 단위셀에 데이터를 재저장하는 동작이 이루어진다.

<47> 메모리 장치의 집적도를 위해 하나의 단위셀을 구성하는 캐패시터는 수펨토(f)정도로 최대한 작게 제조되고, 이로 인해 하나의 단위셀의 캐패시터의 데이터 신호로 저장되는 전하의 양은 매우 작다. 따라서 단위셀의 캐패시터에 저장된 전하가 비트라인에 인가되고, 센스앰프에 의해 증폭된 후에는 다시 증폭된 신호를 가지고 캐패시터에 재저장하는 동작이 반드시 필요하다. 통상적으로 단위셀을 구성하는 캐패시터 충전용량의 90% 이상이 충전되도록 재저장 동작을 수행하고 있다.

<48> 한편, 메모리 장치의 기억소자로 캐패시터를 사용하기 때문에 주기적으로 재충전해주는 리프레쉬 동작이 필요한데, 상기의 재저장 동작을 오래하면 할 수록 리프레쉬 동작의 간격은 길어질 수 있다. 따라서 리프레쉬 동작의 주기를 생각해서는 충분히 재저장동작을 수행해야 한다. 그러나 한번의 명령어에 의해 워드라인을 활성화시키고 나서, 다음 명령어를 입력받아 워드라인을 활성화시키는 시간인 로우사이클 타임 측면에서는 상기의 재저장동작은 작을수록 고속으로 데이터 처리가 가능하기 때문에 재저장동작을 충분히 오랫동안 수행할 수도 없다.

<49> 도3은 도1에 도시된 메모리 장치의 리드 동작에 관한 파형도이다.

<50> 계속해서 도3을 참조하여 살펴보면, 제1 리드명령어(RD0)가 메모리 장치에 입력되면, 첫번째 타이밍(t0)에서 입력된 셀블럭어드레스와 로우어드레스에 대응되는 하나의 셀블럭 및 워드라인(예컨대 셀블럭0(21)의 워드라인(WL0))이 선택되고, 활성화된 워드라인에 대응하는 4K개의 단위셀에 저장된 데이터가 비트라인 센스앰프에 의해 감지, 증폭된다.

<51> 어어서 두번째 타이밍(t1)에서 컬럼어드레스에 의해 디코딩된 컬럼선택신호(CD0,CD1,..)에 의해 4K개의 데이터중에서 선택된 데이터(D0)를 외부로 출력시키고, 다른 한편으로는 활성화된 워드라인에 대응하는 4K개의 데이터에 대한 재저장 동작을 수행한다. 따라서 하나의 리드명령어를 수행하는데 두번의 타이밍(t0,t1)이 필요하다.

<52> 상기의 두구간(t0,t1)이 끝난후에 다시 제2 리드명령어(RD1)가 입력되면 전술한 바와 같이 두 타이밍(t2,t3)동안 한번의 리드동작(RD1)을 수행한다. 도3에 도시된 로우사이클 타임을 로우어드레스를 입력받아 하나의 워드라인이 활성화되고, 다음 로우 어드레스에 의한 워드라인이 활성화될 때까지의 시간을 말한다.

<53> 이어서 데이터를 라이트하는 동작에 대해 간단하게 살펴본다.

<54> 데이터를 저장하는 동작은 데이터를 리드하는 동작에서 설명한 워드라인이 활성화되어 4K개의 데이터가 비트라인 센스앰프에 의해 감지 증폭되는 과정은 똑같다.

<55> 첫번째 타이밍에서 로우어드레스에 의해 하나의 워드라인이 활성화되고, 활성화된 하나의 워드라인에 대응하는 4K개의 데이터가 비트라인 센스앰프에 의해 증폭한다.

<56> 이어서 두번째 타이밍에서 컬럼어드레스에 의해 디코딩된 컬럼선택신호에 해당되는 비트라인 센스앰프에 외부에서 입력된 데이터가 교체 저장되고, 이후에 활성화된 워드라

인에 해당하는 4K개의 데이터를 재저장하는 동작이 수행된다. 이 때에 센스앰프에 교체 저장된 데이터는 외부에서 입력되는 큰 신호이기 때문에 다시 감지 증폭할 필요는 없다. 데이터를 라이트하는 동작도 도3에 도시된 두 타이밍(t0,t1)에 의해 진행된다.

<57> 결국, DRAM 장치의 특성상 데이터를 리드하던지 또는 라이트하던지, 첫번째로 단위 셀에 있는 데이터를 감지, 증폭하는 타이밍과, 두번째로 증폭된 데이터를 외부로 출력하거나 외부에서 입력된 데이터로 교체한 다음, 데이터가 있던 단위셀에 재저장하는 타이밍이 필요한 것이다.

<58> 따라서 하나의 명령어에 따른 데이터를 억세스하고 난뒤에도, 바로 다음데이터를 억세스할 수 있는 것이 아니라 이전 데이터에 대한 재저장동작이 끝난 후에야 다음 데이터 억세스가 가능한 것이다. 즉, 데이터 재저장 시간으로 인해 데이터를 연속으로 억세스할 수가 없는 것이다.

<59> 한편, 보다 고속으로 데이터를 억세스하기 위해 뱅크간 인터리브 모드(Interleave Mode)를 사용하는 메모리 장치이 제안되었다.

<60> 뱅크간 인터리브 모드를 사용하는 메모리 장치이란 일정시간 안에 많은 데이터를 전송하기 위한 뱅크 인터리빙(Bank Interleaving) 방법을 사용되는데, 한 뱅크에서 데이터를 출력하고 재저장하는 순간에 이웃한 뱅크에서 데이터가 연속적으로 출력되어 외부에서 보면 재저장하는 시간없이 연속적으로 명령어를 입력받아 데이터가 출력되는 것으로 보이게 하는 것이다.

<61> 도4에는 뱅크간 인터리빙 동작을 나타내는 파형도이다.

<62> 도4를 참조하여 살펴보면, 인터리빙 동작시에 메모리 장치은 첫번째 리드명령어(RD0)에 의해 뱅크0의 한 워드라인이 활성화되고; 다음 구간에 데이터를 출력시키고 재저장하는 한편, 다른 뱅크1에서 리드명령어(RD1)를 연속적으로 입력받아 한 워드라인을 활성화시키고 데이터(D1)를 출력시킨다. 따라서 도시된 바와 같이, 인터리빙 모드로 동자가게 되면 연속적으로 리드명령어(RD0,RD1,RD2)를 입력받아 데이터(D0,D1,D2)를 연속적으로 출력할 수 있는 것이다.

<63> 그러나, 인터리빙 모드를 사용하는 메모리 장치에서도 동일 뱅크에 연속적으로 억세스가 집중되는 경우에는 인터리빙동작을 수행할 수 없어, 고속으로 데이터를 입출력시킬 수 없는 문제점을 가지고 있다. 입력되는 명령어에 의해 한 뱅크만 연속적으로 데이터를 억세스할 때에는 전술한 뱅크간 인터리빙 모드로 동작할 수 없게 되고, 따라서 입력되는 명령어가 한뱅크의 데이터만을 연속적으로 억세스하는 패턴인지, 또는 뱅크를 옮겨가면서 데이터를 억세스하는 패턴인지에 따라 메모리 장치의 동작속도가 크게 영향을 받게 되는것이다.

<64> 즉, 억세스되는 데이터의 패턴에 따라 메모리 장치의 억세스 속도가 크게 영향을 받게 되는 것이다. 따라서 데이터 억세스 패턴에 관계없이 데이터를 고속으로 억세스할 수 있는 메모리 장치이 필요하다.

【발명이 이루고자 하는 기술적 과제】

<65> 본 발명은 상기의 문제점을 해결하기 위해 제안된 것으로, 고속으로 데이터를 억세스할 수 있는 메모리 장치를 제공하는 것을 목적으로 한다.

<66> 또한 본 발명은 인터리빙모드로 동작하면서도 한 뱅크에서 연속적인 데이터 억세스가 발생될 때도 데이터 억세스 속도의 저하없이 고속으로 동작할 수 있는 메모리 장치를 제공하는 것을 목적으로 한다.

【발명의 구성 및 작용】

<67> 상기의 목적을 해결하기 위한 본 발명은 다수개의 단위셀을 구비하는 제1 셀블럭; 상기 제1 셀블럭에 억세스되는 데이터 재저장을 위한 제2 셀블럭; 및 상기 제1 셀블럭에 대해 연속적으로 제1 데이터 및 제2 데이터가 억세스될 때, 상기 제1 셀블럭에서는 상기 제1 데이터의 재저장동작을 수행하지 않고 상기 제2 데이터가 억세스되도록 제어하고, 상기 제2 셀블럭에서는 상기 제1 데이터의 재저장 동작이 수행하도록 제어하기 위한 제어부를 구비하는 메모리 장치를 제공한다.

<68> 이하, 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자가 본 발명의 기술적 사상을 용이하게 실시할 수 있을 정도로 상세히 설명하기 위하여, 본 발명의 가장 바람직한 실시예를 첨부된 도면을 참조하여 설명하기로 한다.

<69> 도5a는 본 발명에 따른 메모리 장치의 개념을 나타낸 메모리 장치의 블럭구성도이다.

<70> 도5a를 참조하여 살펴보면, 본 실시예에 따른 메모리 장치은 다수개의 단위셀을 구비하는 제1 셀블럭(92)과, 제1 셀블럭(60)에 억세스되는 데이터 재저장을 위한 제2 셀블럭(93)과, 제1 셀블럭(92)에 대해 연속적으로 제1 데이터 및 제2 데이터가 억세스될 때,

제1 셀블럭(92)에서는 상기 제1 데이터의 재저장동작을 수행하지 않고 상기 제2 데이터가 억세스되도록 제어하고, 제2 셀블럭(93)에서 상기 제1 데이터의 재저장 동작이 수행하도록 제어하기 위한 제어블럭(80)을 구비한다.

<71> 또한, 본 실시예에 따른 메모리 장치은 상기 제1 데이터를 제1 셀블럭(60)으로부터 전달받아 래치하기 위한 래치부(91)와, 래치부(91)에 래치된 데이터를 제2 셀블럭(93)으로 전달시키기 위한 신호라인(94)을 더 구비한다.

<72> 또한 제어블럭(80)은 입력되는 논리적 어드레스를 입력받아 물리적 어드레스로 변환하여 출력하기 위한 어드레스 변환부(81)과, 어드레스 변환부(81)의 출력을 입력받아 셀블럭(92,93)의 데이터 억세스를 제어하기 위한 데이터 억세스 제어부(82)를 구비한다

.

<73> 도5b는 본 발명의 개념을 나타내기 위한 메모리 장치의 블럭구성도이며, 특히 셀블럭부분을 나타낸 것이다.

<74> 도5b를 참조하여 살펴보면, 본 발명에 의한 메모리 장치는 제1 및 제2 셀블럭(510c,530c)에 각각 로컬 센스앰프부(510a,510b,520a,520b)를 구비하고 있고, 이웃한 2개의 로컬 센스앰프부(510b,520a) 중 하나를 글로벌 비트라인(720)으로 연결하기 위한 글로벌 비트라인 연결부(610)와, 로컬 센스앰프부(510b,520a)에서 출력되는 데이터를 글로벌 비트라인 연결부를 통해 글로벌 센스앰프부(710)로 전달하는 글로벌 비트라인(720)와, 로컬비트라인 센스앰프부에 의해 감지, 증폭된 데이터를 래치하기 위한 글로벌 센스앰프부(710)를 구비한다. 또한 글로벌 센스앰프부(710)은 글로벌 비트라인(720)을 통과하면서 감소된 데이터신호를 다시 감지, 증폭하는 역할도 하게된다.

<75> 도5c는 바람직한 제2 실시예에 따른 메모리 장치의 블럭구성도이다.

<76> 도5c를 참조하여 살펴보면, 본 실시예에 따른 메모리 장치은 입력되는 어드레스에 대응하는 셀블럭이 8개인 메모리 장치에 있어서, 256개의 워드라인을 각각 구비하며, 예비 워드라인을 상기 256개 만큼 더 구비하기 위해 배치된 9개의 셀블럭(510~590)과, 9개의 셀블럭(510~590) 중에서 선택된 하나의 셀블럭(예컨대 제1 셀블럭(520))에 연속적으로 제1 및 제2 데이터에 대한 억세스가 일어날 때, 선택된 셀블럭(520)에서는 상기 제1 데이터에 대한 재저장동작을 하지 않고 상기 제2 데이터를 억세스하고, 상기 제1 데이터를 억세스하기 위한 워드라인(예컨대 WL0)의 예비 워드라인이 있는 셀블럭에서 상기 제1 데이터의 재저장 동작이 이루어지도록 제어하는 제어부(400)을 구비한다.

<77> 제어부(400)는 9개의 셀블럭(510~590)에 대한 논리적 셀블럭 어드레스(Logical Block Address, LBA)를 저장하기 위한 태그테이블(430)과, 256개의 워드라도5c는 본 발명의 개념을 나타내기 위한 메모리 장치의 블럭구성도.인에 대한 256개의 예비 워드라인 정보를 저장하기 위한 예비블럭 테이블(410)과, 테그테이블(430) 및 예비블럭테이블(410)을 제어하기 위한 태그제어부(420)을 구비한다.

<78> 본 발명은 입력된 셀블럭 어드레스에 대응하는 셀블럭보다 하나를 더 구비하기 때문에 입력되는 셀블럭 어드레스는 논리적인 셀블럭 어드레스로 인식하고, 논리적 셀블럭 어드레스를 해당되는 데이터가 저장된 물리적 셀블럭 어드레스 변환하는 동작이 필요하다.

<79> 제어부(400)는 논리적 셀블럭 어드레스(LBA)(예컨대 논리적 셀블럭0)에 대응되는 물리적 셀블럭 어드레스(예컨대 셀블럭3)로 변환하여 상기 제1 및 제2 데이터가 물리적 셀블럭에 억세스될 수 있도록 한다. 도한 제어부(400)에서는 입력되는 로우어드레스에

의해 선택된 하나의 워드라인에 대응하는 예비블럭을 예비블럭 테이블(410)에서 찾는 동작을 수행한다.

<80> 예를 들어 입력된 로우어드레스에 선택된 워드라인이 셀블럭0의 'WL0'일 때, 워드라인 'WL0'에 대한 예비워드라인이 셀블럭1인 경우에는, 셀블럭0의 'WL0'에 대해서 셀블럭1이 예비블럭이되고, 셀블럭1의 'WL0'이 예비워드라인이 된다. 여기서 찾은 예비 워드라인(셀블럭1의 WL0)은 셀블럭0에 제1 및 제2 데이터가 연속적으로 억세스될 때, 상기 제1 데이터의 재저장동작을 위한 것이다.

<81> 또한 본 실시예에 따른 메모리 장치은 로컬비트라인 센스앰프로 부터 출력되는 데이터를 감지, 증폭하기 위한 글로벌 비트라인 센스앰프(710,730)와 글로벌 비트라인 센스앰프(710,730)와 9개의 셀블럭(510~590)간의 데이터 이동을 위한 글로벌 비트라인(720)과, 글로벌 비트라인(720)과 셀블럭(510~590)을 연결하기 위해 상기 9개의 셀블럭 사이에 구비된 글로벌 비트라인 연결부(610~650)를 구비한다.

<82> 글로벌 비트라인 센스앰프부(710,730)는 9개의 셀블럭(510~590)중에서 셀블럭0(510) 및 셀블럭8(590)의 일측에 2개가 구비되고, 9개의 셀블럭(예컨대 셀블럭0(510))에는 각각 일측과 타측에 로컬센스앰프부(510a,510b)를 2개 구비하고 있다.

<83> 통상적으로 메모리 장치은 다수의 뱅크(예컨대 4 뱅크)를 구비하고 있는데, 각각의 뱅크는 같은 구조를 가지고 있기 때문에, 도5c에는 하나의 뱅크만을 도시하였으며, 또한 도5c에는 본 발명을 설명하기 위한 최소한의 블럭만을 도시하였다.

<84> 하나의 뱅크(1000)에는 총 8개의 세그먼트(1100a ~ 1100h)로 구성되고, 세그먼트(1100a ~ 1100h)에서 출력되는 데이터를 감지, 증폭하여 데이터 입,출력버퍼(3000)로 전

달하거나, 데이터 입,출력버퍼(3000)에서 출력되는 데이터를 세그먼트(1100a ~ 1100h)로 전달하기 위한 I/O 센스앰프부(1200, 1300)를 구비한다.

<85> 또한, 각 세그먼트(1100a ~ 1100h)마다 제어부(400)에서 출력되는 어드레스를 디코딩하여 셀블럭(510~590)으로 출력하기 위한 로우어드레스 디코더부(800)와, 컬럼어드레스를 입력받아 셀블럭(510~590)으로 출력하기 위한 컬럼어드레스 디코더부(900)를 구비한다.

<86> 도5c에 도시된 메모리 장치은 용량이 256Mb인 경우를 나타내고 있는데, 4개의 뱅크로 구성되는 경우 하나의 뱅크에는 64Mb의 단위셀이 구성되며, 하나의 세그먼트(예컨대 1100a)는 8Mb로 구성되며, 총 8개의 세그먼트가 하나의 뱅크를 구성한다. 하나의 세그먼트에는 전술한 바와 같이 9개의 셀블럭(510~590)를 구비하고, 하나의 셀블럭에는 256개의 워드라인(Word Line)과 4K(4 * 1024)개의 비트라인을 구비한다. 이하에서는 하나의 세그먼트가 8Mb이고 9개의 셀블럭이 각각 256개의 워드라인을 구비한 경우에 대해 설명한다.

<87> 도6은 도5c의 도시된 셀블럭 및 센스앰프부중에서 한부분을 나타내는 회로도의 일예로서, 특히 셀블럭0(510)와 셀블럭1(610)과 글로벌 비트라인 연결부(610)의 일부분을 나타낸 것이다.

<88> 도6를 참조하여 살펴보면, 셀블럭0(510)에는 256개의 워드라인(WL0,WL1,...)에 대응하는 하나의 모스트랜지스터와 하나의 캐패시터로 구성된 다수의 단위셀(256 * 4K개)과, 선택된 단위셀의 데이터 신호를 인가받기 위한 로컬 비트라인 (BL0,/BL0,BL1,/BL1,...)과, 비트라인 센스앰프 인에이블신호(RTO, /S)에 의해 인에이블되어 로컬 비트라인에 인가된 데이터 신호를 감지, 증폭하기 위한 로컬 비트라인 센스앰

프(510b_1)와, 로컬 비트라인 센스앰프(520b_1)에 의해 감지, 증폭된 데이터 신호를 글로벌 비트라인 연결부(610)를 통해 글로벌 비트라인(GBL0,/GBL0)으로 전달하기 위한 로컬 센스앰프 연결부(510b_5)를 구비한다.

<89> 또한, 로컬 비트라인 센스앰프부(510b)는 전술한 비트라인 센스앰프(510b_1)뿐만 아니라 로컬 비트라인 센스앰프(510b_1)가 디스에이블일 때에 출력되는 프리차지인에이블신호(BLEQ)에 인에이블되어 비트라인 프리차지 전압(Vblp)으로 비트라인(예컨대 BL0, /BL0)을 프리차지하기 위한 프리차지부(510b_2)와, 이퀄라이제이션 신호(BLEQ')에 의해 셀블럭0(510)의 비트라인(예컨대 BL0,/BL0) 전압레벨을 같게 하기 위한 이퀄라이제이션부(510b_3)와, 글로벌 비트라인 연결부(610)와 로컬비트라인 센스앰프(510b_1)간의 연결을 위한 로컬 비트라인 연결부(510b_4)를 구비하고 있다.

<90> 도6에 도시된 로컬 비트라인 센스앰프부(510b)는 종래의 비트라인 센스앰프부(도2의 123b 참조)와 같은 구성을 하고 있으나, 컬럼제어신호(CD0,..)에 제어되어 비트라인(BL,/BL)의 데이터신호를 I/O 센스앰프부(1200)로 출력하는 데이터 출력부(도2의 123b_5)가 없다. 이는 본 실시예에 따른 메모리 장치에서는 로컬 비트라인 센스앰프(510b)에서 증폭된 데이터신호는 일단 글로벌 비트라인(GBL)을 통해 글로벌 비트라인 센스앰프(도5c의 710,720)로 전달되고, 이후에 글로벌 비트라인 센스앰프(710,720)에서 I/O 센스앰프부(1200)로 출력하기 때문이다.

<91> 또한, 셀블럭1(520)의 로컬비트라인 센스앰프(520a)는 로컬비트라인 연결부(520b_5)를 제외하고 나머지 회로는 도시하지 않았으나, 로컬 비트라인 센스앰프부(510b)와 같은 구성을 가진다. 따라서 본 실시예에 따른 메모리 장치에서는 로컬 비트라인 센스앰프부를 이웃한 셀블럭간(셀블럭0,1) 공유하는 것이 아니라 셀블럭마다 2개의

로컬 비트라인 센스앰프부(예컨대 520a, 520b)를 구비하고 있다. 이로서 종래보다 약간의 면적증가 있으나, 이는 고속 데이터 억세스를 위한 것이다.

<92> 글로벌 비트라인 연결부(610)는 글로벌 비트라인 제어신호(GBIS_01)에 의해 로컬 비트라인 센스앰프부(510b) 또는 로컬 비트라인 센스앰프부(520a)와 글로벌 비트라인(GBL0,/GBL0,GBL2,/GBL2,...)을 연결하며, 글로벌 비트라인(GBL0,/GBL0, GBL2,/GBL2,...)은 글로벌비트라인 센스앰프(710,730;도5c 참조)와 연결된다.

<93> 도6은 셀블럭0,1과 로컬 비트라인 센스앰프의 일부분을 도시한 것으로, 한셀블럭에 4K * 256개의 단위셀을 구비하고 있을 때에 실제로는 256개의 워드라인(WL)과, 4*1024개의 로컬 비트라인(BL) 및 2*1024개의 글로벌 비트라인(GBL)을 구비하게 된다.

<94> 도7은 도5c에 도시된 셀블럭, 글로벌 비트라인 센스앰프부등을 포함하는 셀블럭 영역의 일예를 나타내는 블럭구성도이다.

<95> 도7를 참조하여 살펴보면, 셀블럭영역은 9개의 셀블럭(510~590)과 셀블럭0,9(510)의 일측에 각각 글로벌센스앰프부(710,730)가 구비되어 있고, 글로벌 센스앰프부(710,730)에서 출력되는 데이터를 I/O 센스앰프(1200)로 전달하기 위한 데이터버스(740,750)을 구비하고 있다.

<96> 또한 각 셀블럭(510~590)마다 셀블럭를 제어하기 위한 블럭제어부(510d ~ 590d)가 구비되어 있고, 각 셀블럭의 일측과 타측에 각각 로컬센스앰프(LSA)를 구비되어 있으며, 셀블럭간의 데이터 이동을 위한 글로벌 비트라인(720)과, 셀블럭과 셀블럭사이에 글로벌 비트라인(710)과 로컬 비트라인 센스앰프간의 연결 또는 분리

를 위한 글로벌 비트라인연결부(610~650)를 구비되어 있다. 글로벌 비트라인 연결부(610~650)는 2개의 셀블럭 사이마다 구비되어 셀블럭이 9개일 경우 총 5개가 구비된다.

<97> 입력된 어드레스에 의해 하나의 셀블럭 및 워드라인(셀블럭1(520)의 예컨대 WL0))이 선택되면, 선택된 워드라인(WL0)에 대응되는 4K개의 데이터가 로컬 센스앰프(LSA)에 의해 감지 증폭된다. 감지, 증폭된 데이터는 글로벌 비트라인(720)를 통해 글로벌 비트라인 센스앰프(710,730)로 이동되어 래치된다. 이어진 명령어에서 같은 셀블럭(셀블럭1(520))에서 데이터 억세스가 일어나면, 글로벌 비트라인 센스앰프(710,730)에 래치된 데이터를 글로벌 비트라인(720)을 통해 선택된 워드라인(WL0)에 대한 예비워드라인(WL0)이 있는 셀블럭(예컨대 셀블럭8(590))으로 데이터를 전달하게 된다.

<98> 도8a 및 도8d은 도7에 도시된 메모리 장치에서 셀블럭간 데이터 이동을 보여주는 블럭구성도이다. 여기서는 현재 셀블럭1(520)의 활성화된 워드라인에 해당하는 예비워드라인은 셀블럭8(590)에 있는 경우이다.

<99> 도8a 및 도8b는 셀블럭1(520)에서 셀블럭8(590)로 데이터가 이동되는 것을 도시하고 있다.

<100> 먼저 도8a를 참조하여 살펴보면, 명령어가 수행되는 첫번째 타이밍에서, 입력된 어드레스에 의해 셀블럭1(520)의 워드라인이 활성화되어, 활성화된 워드라인에 따른 데이터가 로컬 비트라인 센스앰프(520a_1,520b_1)에 의해 감지, 증폭된다. 이어서 셀블럭1(520)의 제1 로컬비트라인 센스앰프(520a_1)에 의해 증폭된 데이터가 제1 글로벌 비트라인(A.A')를 통해 글로벌 비트라인 센스앰프(700)로 이동되고, 셀블럭1(520)의 제2 로컬비트라인 센스앰프(520b_1)에 의해 증폭된 데이터가 제2 글로벌 비트라인(B.B')를 통해 글로벌 비트라인 센스앰프(730)로 이동된다.

<101> 이어서 8b를 참조하여 살펴보면, 명령어가 수행되는 두번째 타이밍에서 글로벌 비트라인 센스앰프(710)에 저장되어 있던 데이터가 셀블럭8(590)의 일측에 구비된 로컬 비트라인 센스앰프(590a_1)로 이동되고, 글로벌 비트라인 센스앰프(730)에 저장되어 있던 데이터는 셀블럭8(590)의 로컬 비트라인 센스앰프(590b_1)로 입력된다. 이 타이밍에서 리드명령어를 수행중일 때는 글로벌 비트라인 센스앰프(710,730)로부터 데이터가 I/O 센스앰프부로 출력되고, 라이트명령어를 수행중일 때는 데이터가 I/O 센스앰프부로부터 글로벌 비트라인 센스앰프(710,730)로 입력된다.

<102> 도8c 및 도8d는 데이터가 셀블럭7(580)에서 셀블럭0(510)으로 이동되는 것을 도시하고 있다. 여기서는 현재 셀블럭7(580)의 활성화된 워드라인에 해당하는 예비워드라인은 셀블럭0(510)에 있는 경우이다.

<103> 명령어가 수행되는 첫번째 타이밍에서 도8c에 도시된 바와 같이 셀블럭7(580)의 로컬 비트라인 센스앰프(580a_1,580b_1)에서 글로벌 비트라인 센스앰프(710,730)로 이동되고, 두번째 타이밍에서 도8d에 도시된 바와 같이, 글로벌 비트라인 센스앰프(710,730)에 저장된 데이터가 셀블럭0(510)의 로컬 비트라인 센스앰프(510a_1,510b_1)로 이동된다.

<104> 여기서 글로벌 비트라인 연결부(610~650)는 2개의 셀블럭당 하나만 구비되고, 적절히 제어함으로서, 한 셀블럭의 로컬 비트라인 센스앰프부(예컨대 520a,520b)와 글로벌 비트라인 센스앰프(710,730)간에 데이터가 이동될 수 있다.

<105> 도9 내지 도11은 본 발명에 의한 메모리 장치에서 동작을 나타내는 파형도이다.

<106> 도9는 서로 다른 셀블럭으로 데이터를 억세스할 경우, 인터리브 동작을 보여주는 파형도이다. 또한, 도10는 하나의 셀블럭에서 데이터를 연속적으로 리드할 때의 동작을

보여주는 파형도이며, 도11은 하나의 셀블럭에 데이터를 연속적으로 라이트할 때의 동작을 보여주는 파형도이다.

<107> 도9를 참조하여 본실시예에서의 메모리 장치에서 인터리브모드에서의 리드동작을 살펴본다.

<108> 먼저, 본 발명에 의한 메모리 장치에 입력되는 어드레스에 대해 살펴보면, 입력되는 어드레스는 뱅크 및 세그먼트와, 한 세그먼트내에서 하나의 셀블럭을 지정하기 위한 셀블럭어드레스(BA)와, 선택된 셀블럭내에서 하나의 워드라인을 선택하기 위한 로우어드레스(RA)와, 비트라인을 선택하기 위한 컬럼어드레스(CA)로 구성된다. 이 때 입력되는 셀블럭어드레스(BA)에 대응되는 셀블럭은 한 세그먼트에서 8개이고, 여분의 하나는 예비블럭으로 된다.

<109> 예비블럭은 각 워드라인에 따라 다르게 정해지며, 이를 저장하고 있는 것이 예비블럭 테이블이다. 따라서 입력되는 셀블럭어드레스는 논리적 셀블럭 어드레스이고, 실제 해당되는 데이터가 저장되어 있는 물리적 셀블럭 어드레스로 변환하는 과정이 명령어 실행시 마다 필요하다. 셀블럭 어드레스의 변환 과정에 대해서는 자세히 후술한다.

<110> 이전명령에 대한 비트라인에 인가된 데이터정보를 리셋하기 위해 비트라인을 프리차지하는 동작이 수행된다. 이어서 리드명령어(RD0) 입력되면, 첫번째 타이밍(t0)에서 입력된 논리적 셀블럭 어드레스(LBA)를 물리적 셀블럭 어드레스(PBA)로 변환하여 하나의 셀블럭을 선택한다. 이어서 선택된 셀블럭에서 로우어드레스에 의해 하나의 워드라인이 활성화된다. 활성회된 워드라인에 해당하는 4K개의 단위셀 데이터가 로컬 비트라인(BL,/BL)에 인가되고 로컬 비트라인 센스앰프에 의해 감지, 증폭된다. 증폭된 데이터는

글로벌 비트라인(GBL,/GBL)을 통해 글로벌 비트라인 센스앰프(도5c의 700,700')로 이동이 된다. 이 과정을 '프리액티브(pc_act) 과정'이라고 정의한다.

<111> 이어서 두번째 타이밍(t1)에서 글로벌 비트라인 센스앰프(700,700')에 저장된 4K개의 데이터중에서 컬럼선택신호(CD0,CD1,...)에 의해 선택된 데이터가 외부로 출력되는 한편, 증폭된 4K개의 데이터는 원래의 단위셀로 재저장된다. 따라서 한번의 리드동작에 두단계가 필요 하다.

<112> 상기의 두번째 타이밍(t1)에서 셀블럭0으로부터 데이터(D0)가 출력되고 있을 때, 다음 리드명령어(RD1)의 프리액티브(pc_act) 과정이 수행되어 셀블럭1에서의 하나의 워드라인이 활성회되고, 그에 해당하는 4K개의 데이터라 로컬 비트라인 세스앰프에 의해 감지, 증폭된다.

<113> 다음 타이밍(t2)에서 셀블럭1에서 리드명령어(RD1)에 의한 데이터(D1)가 출력되고 재저장되는 한편, 셀블럭0에서는 다음 리드명령어(RD2)를 입력받아 프리액티브 동작을 수행한다.

<114> 따라서 서로 다른 블럭으로 데이터를 억세스할 때에는 하나의 블럭에서 데이터를 재저장하는 동안 다른 블럭에서 프리액티브 동작을 수행하는 인터리빙동작을 수행하고, 이로 인해 도14에 도시된 바와 같이 연속적으로 데이터를 출력할 수 있다. 도14에 도시된 블럭간 인터리빙에 의한 로우사이클 타임은 한번의 명령어 입력으로 워드라인이 활성화되고 다시 다음명령어로 인한 워드라인이 활성화되는 기산을 말한다.

<115> 이어서 도10를 참조하여, 하나의 셀블럭에 데이터를 연속적으로 리드할 때의 동작을 살펴본다.

<116> 도10에 도시된 바와 같이 총 8번의 리드명령어(RD0~RD7)가 입력되는 것으로 가정하고, 각각의 명령어에 표시된 괄호안의 내용은 논리적 셀블럭 어드레스와 로우어드레스에 따른 워드라인을 나타낸다. 또한 논리적 셀블럭 어드레스가 'BL0'인 경우 물리적 셀블럭도 셀블럭0인 것으로 가정하고, 상기의 8번의 리드명령어(RD0~RD7)에 해당되는 예비블럭은 블럭1인 것으로 가정한다.

<117> 첫번째 타이밍(t0)에서 첫 리드명령어(RD0)가 입력되면, 입력되는 논리적 셀블럭어드레스(BL0)를 물리적 셀블럭어드레스로 변환한다. 여기서는 로우어드레스에 따른 워드라인(WL0)에 대응하는 논리적 셀블럭 어드레스(BL0)의 물리적 셀블럭 어드레스는 셀블럭0이다. 도15에서 명령어실행 타이밍에서 빗금친부분이 논리적 셀블럭어드레스를 물리적 셀블럭 어드레스로 변환하는 구간을 나타낸다.

<118> 변환된 물리적 셀블럭 어드레스로 인해 셀블럭0가 선택이 되고, 로우어드레스에 따라 워드라인(WL0)가 활성화 된다. 이어서 활성화된 워드라인(WL0)에 대응하는 4K개의 단위셀 데이터가 로컬 센스앰프에 의해 감지 증폭된다. 이어서 로컬 센스앰프에 의해 감지 증폭된 4K개의 데이터는 글로벌 비트라인을 통해 글로벌 센스앰프(700,700')로 이동된다. 즉 첫번째 타이밍(t0)에서 리드명령어(RD0)의 프리액티브과정을 수행하는 것이다. 다른한편으로 첫번째 타이밍(t0)에서 테그제어부(200)에 의해 워드라인(WL0)에 해당하는 예비블럭이 셀블럭1 임을 감지하고 셀블럭1에 해당하는 태그1을 업데이트 한다.

<119> 이어서 다음 리드명령어(RD1)가 물리적 셀블럭0에 관한 것이므로, 리드명령어(RD0)에 관한 재저장동작을 수행하지 않고 바로 리드명령어(RD1)에 관한 프리액티브 과정을 두번째 타이밍(t1)에서 수행한다.

<120> 한편으로 두번째 타이밍(t1)에서 글로벌 센스앰프로 저장되어 있던 4K개의 데이터 중에서 컬럼어드레스에 의해 선택된 데이터는 I/O 센스앰프로 거쳐 외부로 출력된다. 또한 이 타이밍(t1)에서 글로벌센스앰프에 저장된 4K개의 데이터- 리드명령어(RD0)에 의해 재저장될 4K개의 데이터-는 예비블럭으로 지정된 셀블럭1로 이동이 된다. 데이터이동은 글로벌 비트라인(GBL,/GBL)을 통해 이동된다. 도15에 도시된 'intw0'가 상기의 과정을 나타낸다.

<121> 이제부터는 물리적 셀블럭1의 워드라인(WL0)이 논리적 셀블럭0의 워드라인(WL0)이 되고, 물리적 셀블럭0이 워드라인(WL0)에 대한 예비블럭으로 지정되는데, 이에 따라 예비블럭 테이블(410)의 저장정보를 업데이트 한다.

<122> 계속해서 상기와 같은 방법으로 리드명령어(RD1)를 입력받아 첫번째 타이밍(t1)에서 프리액티브과정을 수행하고, 두번째 타이밍(t2)에서 데이터를 예비블럭(블럭1)으로 이동시켜 재저장하면서 데이터를 출력한다. 이 때에도 각각의 타이밍에서 예비블럭 테이블(410) 및 태그테이블(430)를 업데이터 하게 된다. 도15에 도시된 'intw1'가 상기의 과정을 나타낸다. 이어서 리드명령어(RD2)를 입력받아 첫번째 타이밍(t2)와 두번째 타이밍(t3)에 전술한 바와 같은 동작을 수행한다. 이 때에도 각각의 타이밍에서 예비블럭테이블(430) 및 태그테이블(430)을 업데이트 하게 된다. 도15에 도시된 'intw2'가 상기의 과정을 나타낸다.

<123> 이어서 리드명령어(RD3)이 입력되어 리드동작을 수행할 때에는, 다음 리드명려어(RD4)에 해당되는 물리적 셀블럭이 리드명령어(RD0)에 바뀌어서 이제는 셀블럭1이기 때문에, 리드명령어(RD3)에 관한 프리액티브 동작후 데이터를 예비블럭으로 이동하여 재저장시킬 필요가 없다. 따라서 이 타이밍(t3)에서는 셀블럭0에서 프리액티브 동작을 수행

하고, 타이밍(t4)에서는 데이터를 출력하고, 셀블럭0에서 워드라인(WL3)에 관한 4K개의 단위셀로 재저장동작을 수행한다. 이때에는 예비블럭테이블(430)을 업데이터할 필요가 없다.

<124> 계속해서 리드명령어(RD4~RD5)이 입력되면, 전술한 바와 같이 첫타이밍에서 프리액티브동작을 하고, 다음 타이밍에서 예비블럭으로 데이터를 이동시켜 재저장동작을 수행하고, 리드명령어(RD6,RD7)는 인터리브동작으로 리드명령어(RD3)에서와 같이 각각 셀블럭0 및 셀블럭1에서 프리액티브 동작 및 데이터 재저장동작을 수행한다.

<125> 도10에 도시된 점선부분은 실제 입력된 리드명령어에 따른 동작이 수행되고 있지만 내부적으로는 해당되는 데이터는 예비블럭으로 이동되고, 예비블럭테이블 (410)및 태그테이블(430)이 업데이트되는 구간이다. 또한, 'EBT_UP'는 테그테이블을 업데이트하기 위한 신호이고, 'EBT_UPN'는 테그 테이블을 업데이트 하지 않도록 하는 신호이다. 또한 'X'구간이 논리적 셀블럭 어드레스를 물리적 셀블럭 어드레스로 변환하는 시간이다.

<126> 결론적으로, 같은 셀블럭에 연속적으로 데이터를 리드할 경우에도 첫번째 타이밍에서 하나의 워드라인을 활성화시키고, 해당되는 데이터를 증폭시킨다음 글로벌비트라인 센스앰프로 이동시키는 프리액티브 동작을 수행하고, 다음 타이밍에서 데이터를 출력할 때 동일한 셀블럭에서 재저장하지 않고 글로벌 비트라인센스앰프와 글로벌 비트라인을 이용해서 예비블럭을 데이터를 옮겨서 재저장하기 때문에, 연속적으로 리드명령어를 입력받아 데이터를 출력시킬 수 있는 것이다.

<127> 이어서 도11를 참조하여 하나의 셀블럭에서 데이터를 연속적으로 라이트할 때의 동작을 살펴본다.

<128> 먼저, 라이트명령어(WR0)가 어드레스와 함께 입력되면, 입력되는 논리적 셀블럭 어드레스(BL0의 WL0)를 해당되는 물리적 셀블럭 어드레스(BL1)로 변환시킨다.첫번째 타이밍(t0)에서 셀블럭1의 워드라인(WL0)을 활성화시키고, 활성화되는 워드라인(WL0)에 대응되는 4K의 데이터를 증폭하고, 글로벌비트라인 센스앰프로 이동시키는 프리액티브 동작을 수행한다.

<129> 두번째 타이밍(t1)에서 글로벌비트라인 센스앰프에 이동딘 4K개의 데이터중에서 컬럼어드레스에 해당되는 데이터를 입력되는 데이터로 교환하고, 셀블럭1의 워드라인(WL0)에 해당되는 4K개의 단위셀에 재저장시킨다. 따라서 데이터를 증폭한 뒤에 외부에서 입력된 데이터로 교환하는 동작만 제외하고는 리드동작과 같은 방식으로 라이트동작이 진행된다.

<130> 이어서, 두번째 타이밍(t1)에서도 라이트명령어(WR1)의 프리액티브 동작을 논리적 셀블럭 어드레스(BL0,WL1)에 대응되는 셀블럭2에서 수행하고, 세번째 타이밍(t2)에서는 외부에서 입력된 라이트될 데이터(D1)를 해당되는 센스앰프로 대체시키고난 후, 4K개의 데이터를 셀블럭2의 워드라인(WL1)에 해당되는 4K개의 단위셀에 저장한다.

<131> 다른 한편으로 세번째 타이밍(t2)에서 라이트명령어(WR2)의 프리액티브 동작을 논리적 셀블럭 어드레스(BL0.,WL2)에 대응되는 셀블럭1에서 수행하고, 네번째 타이밍(t3)에서는 프리액티브 동작시 증폭된 4K개의 데이터가 셀블럭1에 저장되지 않고, 예비블럭으로 지정된 셀블럭0로 이동하여 저장된다. 이 때에 예비블럭태이블(410) 및 태그테이블(430)이 업데이트 된다. 이는 다음 라이트 명령어(WR3)에 억세스되는 셀블럭도 셀블럭1이기 때문에, 타이밍(t2)에 예비블럭으로 지정된 셀블럭0로 라이트명령어

(WR2)에 대한 재저장될 4K개의 데이터가 셀블럭0로 이동(intW0)되는 것이다. 이 때에 태그테이블업데이터가 발생한다. 도11에 도시된 'intw0'가 상기의 과정을 나타낸다.

<132> 이후에 라이트명령어(WR4,WR5)가 수행되는 동안에도 연속해서 같은 셀블럭을 억세스하기 때문에, 상기와 같은 동작으로 타이밍(t5)에 내부적으로 데이터가 예비블럭(셀블럭1)으로 이동되어 데이터 재저장 동작이 수행된다. 도16에 도시된 'intw1'가 상기의 과정을 나타낸다.

<133> 도11의 명령어 실행 타이밍에서 점선부분은 입력되는 라이트명령어가 수행중이기는 하나 다음 라이트 명령어가 같은 셀블럭을 억세스하기 때문에 데이터가 예비블럭으로 이동되어 재저장 동작이 일어나는 구간을 나타내고 있다.

<134> 따라서, 본 실시예에 따른 메모리 장치은 서로 다른 셀블럭으로 데이터를 억세스할 경우에는 인터리빙 방법을 사용하여 데이터를 억세스하고, 같은 셀블럭으로 데이터를 억세스할 경우에는 추가로 구비된 예비블럭으로 억세스된 데이터가 이동되어 재저장 동작이 수행되는 것이다. 이는 각 세그먼트마다 실제 사용되는 블럭보다 하나의 셀블럭을 더 추가함으로 가능한 것이고, 이렇게 데이터를 억세스하기 때문에 데이터가 억세스되는 패턴에 상관없이 항상 고속으로 데이터를 억세스할 수 있는 것이다.

<135> 도12은 본 발명에 의한 메모리 장치에서 도10에 도시된 순서대로 리드명령어(RD0~RD7)가 입력될 때의 시뮬레이션 파형도이다.

<136> 도10을 참조하여 살펴보면, 리드명령어(RD0~RD2,RD4,RD5))가 실행될 때에는 다음 실행 명령어가 계속 같은 셀블럭를 억세스하기 때문에, 억세스된 데이터의 재저장동작은 예비블럭으로 지정된 셀블럭1에서 수행되어, 총 5번의 데이터 이동(intW)이 일어나고,

리드명령어(RD3,RD6,RD7)가 실행될 때에는 다음 실행명령어가 다른 셀블럭을 억세스하기 때문에, 선택된 하나의 블럭에서 3번의 프리액티브 및 데이터 재저장동작(act)이 일어나는 것을 알 수 있다.

<137> 한편, 본 발명에서는 입력되는 셀블럭 어드레스를 논리적 셀블럭 어드레스로 취급하고, 이를 실제 데이터가 저장되어 있는 물리적 셀블럭 어드레스로 변환하여 데이터 억세스 동작을 진행한다. 따라서 논리적 어드레스를 물리적 셀블럭 어드레스로 변환하기 위한 장치가 필요하며, 또한 현재 명령어에 의해 활성화되는 워드라인에 대한 예비워드라인을 찾기 위한 장치가 필요하며, 상기의 동작을 수행하는 것이 5b에 도시된 제어부(400)를 구성하는 예비블럭 테이블(410), 테그제어부(420), 테그테이블(430)등이다.

<138> 도13은 도5c에 도시된 태그제어부(400)를 나타내는 블럭구성도이다.

<139> 도13을 참조하여 살펴보면, 테그제어부(420)는 외부에서 입력되는 명령어(Ext_CMD), 논리적 셀블럭어드레스(Ext_BA), 로우어드레스(Ext_RA)를 입력받아 현재 클럭에서의 실행 명령어(Cur_CMD), 논리적 셀블럭 어드레스(Cur_LBA), 로우 어드레스(Cur_RA)와, 이전 클럭에서의 실행 명령어(Pre_CMD), 논리적 셀블럭 어드레스(Pre_LBA), 로우어드레스(Pre_RA)를 출력하기 위한 신호입력부(421)와, 현재 클럭에서의 실행 명령어(Cur_CMD), 논리적 셀블럭 어드레스(Cur_LBA), 로우어드레스(Cur_RA)와, 이전 클럭에서의 실행 명령어(Pre_CMD), 논리적 셀블럭 어드레스(Pre_LBA), 로우어드레스(Pre_RA)를 입력받아, 이를 서로 비교하여 태그테이블(300) 및 예비블럭 테이블(100)을 제어하기 위한 어드레스 변환 제어부(422)로 구성된다.

<140> 여기서 어드레스 변한 제어부(422)에서 출력되는 예비블럭 업데이터 신호(EBT_UPDATE)는 이전 클럭과 현재클럭에 억세스할 물리적 셀블럭어드레스(Pre_RA,Cur_LBA)가 같을 경우, 예비블럭 테이블(100)에 저장된 내용을 업데이터하기 위한 신호이고, 프리차지 액티브신호(pc_act[0:8])는 입력되는 논리적 블럭 어드레스에 대응하는 데이터가 저장되어 있는 물리적 셀블럭어드레스를 선택하기 위한 신호이다. 또한 내부 재저장 신호(intW)는 이전 클럭과 현재클럭에 억세스할 물리적 블럭어드레스(Pre_LBA,Cur_LBA)가 같을 경우, 현재 셀블럭에서 억세스한 데이터를 예비블럭으로 선택된 셀블럭에 재저장하기 위한 신호인데, 하나의 명령어 수행중 두번째 타이밍에 실행하기 위해 클럭지연부(422_1)를 거쳐서 출력이 된다.

<141> 또한, 명령어 취소신호(CMD_KILL)는 연속해서 동일한 셀블럭의 워드라인을 억세스할려고 할 때에는 뒤에 실행되는 명령어의 동작을 중지시키기위한 신호이다. 이는 연속해서 같은 셀블럭의 같은 단위셀을 억세스하는 경우에 앞 명령어에 대한 태그가 제대로 업데이터 되지 않은 상태에서 같은 태그 정보를 읽게되면 오류가 생기기 때문이다. 이때에는 나중의 명령어에 관한 동작을 취소시키고, 이전의 데이터를 그대로 억세스하면 되는 것이다.

<142> 도14는 도5c에 도시된 예비블럭 테이블(410)을 나타내는 블럭구성도이다.

<143> 도14를 참조하여 살펴보면, 예비블럭 테이블(410)에는 하나의 셀블럭에 총 256개의 워드라인이 있기 때문에, 256개의 워드라인에 대한 예비블럭 정보를 저장하기 위해 256개의 레지스터가 있고, 각각의 레지스터는 한 세그먼트에 구비된 9개의 블럭에 대한 정보를 저장해야 하기 때문에 각각 4비트를 저장할 수 있게 구성된다. 예비블럭 테이블

(410)에 저장된 내용을 살펴보면, 워드라인(WL0)의 예비워드라인은 셀블럭1에 있다는 뜻 것이고, 워드라인(WL3)의 예비워드라인은 셀블럭4에 있다는 뜻이이다. 메모리 장치의 동작중에 예비블럭테이블(410)에 저장된 내용을 계속 업데이트되며, 업데이트 될 때 마다 256개의 워드라인에 대한 256개의 예비워드라인에 대한 정보는 계속 바뀔수 있다.

<144> 예비블럭 테이블(410)은 현재 명령어에 대한 로우어드레스(Cur_LA)를 입력받아 태그제어부(420)로 현재 실행되고 있는 로우어드레스(Cur_LA)에 대한 예비블럭을 알려주는 신호(Extra_BA)를 출력하고, 예비블럭 업데이터 신호(EBT_UPDATE)에 따라 이전 클럭의 로우 어드레스(Pre_RA)를 입력받아 예비블럭정보(Pre_PBA)를 업데이트 한다.

<145> 도15은 도5b에 도시된 태그테이블(430)을 나타내는 블럭구성도이다.

<146> 도15을 참조하여 살펴보면, 태그테이블(430)은 한 세그먼트(1100a)에 구비된 9개의 셀블럭에 대한 논리적인 셀블럭 어드레스를 저장하기 위해 9개의 태그 메모리(432a~432i)와, 예비블럭신호(Extra_BA)를 디코딩하여 9개의 태그메모리(432a~432i)로 출력하는 셀블럭 어드레스 디코더부(431)와, 9개의 태그메모리(432a~432i)에서 출력되는 논리적 셀블럭 어드레스(LBA)와 현재 클럭의 셀블럭어드레스(Cur_LBA)를 비교하기 위한 9개의 비교부(433a~433i)와, 9개의 비교부(433a~433i)에서 출력되는 신호를 앤코딩하여 현재 억세스될 물리적 셀블럭 어드레스(Cur_PBA)를 출력하기 위한 셀블럭어드레스 앤코더부(434)와, 현재클럭의 물리적 셀블럭 어드레스(Cur_PBA)를 한클럭 지연시켜 이전 클럭의 물리적 셀블럭 어드레스(Pre_PBA)를 출력하기 위한 태그지연부(535)을 구비한다.

<147> 여기서 각각의 태그메모리(432a~432i)는 256개의 레지스터를 구비하고, 각각의 레지스터는 8개의 논리적 셀블럭 어드레스를 저장하기 위해 3비트로 구성된다. 또한, 각각의 태그메모리(432a~432i)에서 제1 레지스터(0)는 워드라인'WL0'의 논리적 셀블럭어드레

스를 저장하고 제2 레지스터(1)는 워드라인'WL1'의 논리적 셀블럭 어드레스를 저장하고, 제256 레지스터(255)는 워드라인 'WL255'의 논리적 셀블럭 어드레스를 저장한다.

<148> 예컨대 제1 태그메모리(432a)를 참조하면, 물리적 셀블럭0의 워드라인(WL0)은 논리적 셀블럭1의 워드라인(WL0)을 가리키고 있고, 물리적 셀블럭0의 워드라인(WL255)은 논리적 셀블럭7의 워드라인(WL255)을 가리키고 있는 것이다.

<149> 도16은 도15에 도시된 9개의 테그메모리(432a~432i)의 일예를 나타내는 회로도이고, 도17는 도14에 도시된 예비블럭테이블의 일예를 나타내는 회로도이다.

<150> 도16에 도시된 바와 같이, 통상적인 셀블럭에 구비된 단위셀을 이용하여 256*3비트의 저장용량을 가지는 테그메모리를 9개 구비할 수 있다. 한편, 예비블럭 테이블도 도16에 도시된 바와 같은 셀블럭의 단위셀 구조를 이용해서 구성할 수 있으나, 예비블럭테이블은 한클럭내에서 테그내의 정보를 읽고, 다시 업데이터를 해야하기 때문에 도17에 도시된 바와 같이 고속억세스가 가능한 스태택셀로 구성할 수 있다.

<151> 도18은 본 발명에 의한 메모리 장치의 태그관련 블럭의 동작을 나타내는 파형도이다.

<152> 이하에서는 도15 내지 도18을 참조하여 태그관련 블럭의 동작에 대해서 설명한다.

<153> 하나의 명령을 수행하기 위해, 외부에서 어드레스가 입력되면, 먼저 예비블럭테이블(410)에서 현재 입력된 로우어드레스에 해당되는 워드라인(예컨대 WL0)의 예비워드라인(WL00)이 어떤 셀블럭에 있는지 감지된다. 예를 들어 도14에 도시된 바에 의하면 워드라인 'WL0'에 대한 예비워드라인은 셀블럭1에 있는 것으로 되는 것이다. 여기서 감지된 셀블거1에 대한 셀블럭 어드레스는 물리적 셀블럭어드레스이다.

<154> 이어서 태그테이블(430)의 셀블럭어드레스 디코더부(431)에서는 예비블럭 테이블에서 감지된 예비 셀블럭 어드레스(Extra_BA)를 디코딩하여 9개의 태그메모리중 하나를 선택하고, 선택된 태그메모리(예컨대 432b)의 제1 레지스터(0)에 현재 로우어드레스(Cur_RA)를 저장한다.

<155> 한편 9개의 태그메모리(432a~432i)에서는 현재 로우어드레스(Cur_RA)에 따라 정해지는 워드라인(예컨대 'WL0')에 따라 제1 레지스터에 저장된 논리적 셀어드레스를 각각 출력한다.

<156> 이어서 현재 논리적 셀블럭 어드레스(Cur_LBA)와 9개의 비교부(433a~433i)에서 각각 비교된다. 예를 들어 로우어드레스에 대응하는 워드라인이 'WL0'이고, 논리적 셀블럭 어드레스(Cur_LBA)가 셀블럭1에 대한 어드레스이면, 첫번째 비교부(433a)의 출력신호가 인에이블되어 셀블럭 어드레스 인코더부(434)로 출력되고, 이로 인해 논리적 셀블럭1에 관한 어드레스는 물리적 셀블럭0에 관한 어드레스로 변환되는 것이다.

<157> 이어서 셀블럭 어드레스 인코더부(434)는 9개의 비교부에서 출력되는 신호를 인코딩하여 현재 물리적 셀블럭 어드레스(Cur_PBA)를 출력하고, 태그지연부(435)에서는 클럭신호를 입력받아 현재 물리적 셀블럭 어드레스(Cur_PBA)를 한클럭 지연시킨 이전 물리적 셀블럭 어드레스(Pre_PBA)를 출력한다.

<158> 도18에는 상기의 동작을 나타내는 파형도가 도시되어 있는데, 하나의 명령어가 수행되는 제1 타이밍에서 입력된 예비블럭 테이블(410)에서 로우어드레스에 따라 선택되는 하나의 워드라인(예컨대 WL0)에 대응하는 예비블럭(셀블럭1(BL2))을 찾고(A), 태그테이블(430)에서 입력된 논리적 셀블럭 어드레스(Cur_LBA)를 물리적 셀블럭어드레스

(Cur_PBA)로 변환하고(B), 이와 동시에 현재 입력된 논리적 셀블럭 어드레스를 찾은 예비블럭에 저장한다.(C)

<159> 이어서 명령어가 수행되는 제2 타이밍에 다음 실행될 물리적 셀블럭어드레스와 현재 실행되고 있는 물리적 셀블럭어드레스를 비교하는데(D), 비교한 결과 같은 물리적 셀블럭에 대한 억세스이면, 예비블럭 테이블(EBT)를 업데이트 하게 된다(E). 예비블럭 테이블(EBT)가 업데이트되면 상기 'C'단계에서 저장된 내용에 따라 예비블럭이 바뀌는 것이다.

<160> 도18에 도시된 f,g 단계는 이전 명령어에 대한 비교 및 예비블럭 테이블(EBT)에 관한 동작이고, h,i,j는 제2 타이밍부터 실행될 다음 명령어에 관한 동작이다. 따라서 명령어가 실행되는 매 타이밍마다 예비블럭 테이블(410)에서 예비블럭을 찾아서 현재의 논리적 셀블럭 어드레스(Cur_LBA)를 저장하고, 논리적 셀블럭 어드레스(Cur_LBA)를 물리적 셀블럭 어드레스(Cur_PBA)로 변환하는 한편, 현재 타이밍에 억세스된 셀블럭어드레스와 다음 타이밍에 억세스될 셀블럭 어드레스를 비교하여 같으면 예비블럭 테이블(EBT)를 업데이트한다.

<161> 도19은 도6 또는 도7에 도시된 글로벌 비트라인 연결부를 제어하기 제어부의 일예를 나타내는 회로도이다.

<162> 도19에 도시된 부분은 본발명에 의해 추가로 구비된 글로벌 비트라인 연결부(610~650)를 제어하기 위한 회로로서, 도6에 도시된 바와 같이 종래의 셀블럭 제어부에 구비될 수도 있고, 글로벌 비트라인 연결 제어부를 따로 구성할 수 있다.

<163> 도20는 도19에 도시된 제어부에 따라 생성되는 제어신호에 따라 본 발명의 메모리 장치 동작을 보여주는 파형도이다.

<164> 도20에 도시된 빗금친 부분은 각 타이밍별로 논리적 셀블럭 어드레스를 물리적 셀블럭 어드레스로 변환하는 구간이며, 도13에 도시된 딜레이에 따라 생기는 지연시간($\tau 1, \tau 2, \tau 3$)에 따라 예비셀 블럭과 노멀셀블럭의 글로벌 비트라인 연결부의 연결신호의 파형이 정해진다. 또한 도19에서 △는 명령어 실행시의 첫타이밍에서 활성화된 워드라인을 다음 명령이 실행시 같은 셀블럭을 억세스할 경우 강제로 비활성화시키는 부분을 나타낸 것이고, 'intW' 명령어실행시의 두번째 타이밍에서 예비블럭으로 데이터가 이동되어 재저장되는 것을 나타낸다. 또한 'LSA_EN'은 로컬 비트라인 센스앰프의 인에이블신호를 나타내는 것이고, 'BIS'는 로컬 비트라인 연결부의 인에이블신호이고, 'GBIS'는 글로벌 비트라인 센스앰프의 인에이블 신호이고, 'GSA_EN'은 글로벌 비트라인 센스앰프의 인에이블 신호이다.

<165> 전술한 실시예에서는 다음 명령어 수행시 억세스 될 셀블럭이 이전의 명령어 수행시 억세된 셀블럭과 같을 때만 데이터를 이동시키고, 데이터가 이동시에만 태그를 업데이트 해주는 방법을 사용하였다.

<166> 또한, 현재 억세스될 셀블럭에 워드라인을 활성화 시키는 동시에 선택된 예비 셀블럭으로 데이터를 항상 이동시켜 재저장시키는 방법으로 본 발명의 사상을 구현할 수 있다. 이 때에는 항상 두군데의 셀블럭에 같은 데이터가 저장되며, 실제의 데이터가 저장된 셀블럭은 태그테이블에 의해 정해진다. 상기의 두가지 방법은 메모리 장치의 설계하는 방법에 따라 적절하게 선택될 수 있으며, 어느 방법으로 메모리 장치을 구현하던지 데이터 재저장하는 시간과 상관없이 데이터를 고속으로 억세스할 수 있다.

<167> 본 발명의 사상을 구현하려고 하면, 종래의 메모리 장치구조에서 하나의 예비 셀블럭과, 예비블럭 테이블, 태그테이블 및 테그 제어부가 추가로 구비되어야 한다. 또한 글로벌 비트라인 센스앰프와 글로벌 비트라인이 추가로 구비되어야 하기 때문에 통상적인 메모리 장치보다 추가적인 면적이 필요하다. 그러나 상기의 블럭을 추가함으로서 메모리 장치의 동작속도는 재저장하는 시간만큼 감소되어, 종래의 메모리 장치보다 고속으로 데이터를 억세스 할 수 있다.

<168> 종래 기술에 의해 메모리 장치의 로우사이클 타임은 tRC라고 생각했을 때, 본 발명에 따른 메모리 장치의 로우 사이클 타임은 MAX{0.5*(tBAT,+tRP+tRC),tINTW}로 정해진다. 여기서 tBAT는 셀블럭어드레스 변환시간이고, tRP는 프리차지 타임이며, tRC는 종래의 로우사이클 타임이고, tINTW는 내부적으로 지정된 예비블럭으로 데이터가 이동되어 재저장하는 시간을 나타낸다. 여기서 tRP는 본발명의 사상을 구현하기 위해 하나의 명령어가 수행될 때마다 하는 강제프리차지 동작으로, 연속해서 같은 셀블럭을 억세스할 때 다음 데이터를 바로 억세스하기 위해 이전에 감지된 데이터를 제거하기 위해 로컬 비트라인 센스앰프를 프리차지시키는 것을 말한다.

<169> 본 발명에 따른 메모리 장치의 로우사이클 타임은 0.5*(tBAT,+tRP+tRC)와 tINTW 중에서 소요되는 시간이 더 긴 타이밍으로 정해지는데, '0.5*(tBAT,+tRP+tRC)' 시간이 tINTW 시간보다는 더 많은 시간이 걸려서, 로우사이클 타임은 0.5*(tBAT,+tRP+tRC)로 정해진다.

<170> 따라서 논리적 셀블럭어드레스를 물리적 셀블럭어드레스로 변환하는 시간과 강제프리차지하는 시간이 추가되더라도 데이터 재저장하는 만큼 감소되어 대략 30~40%의 로우사이클 타이밍이 절감된다. 예를 들어 명령어하나가 수행되는 타이밍이 15n라고 생각

하면, 종래에는 두번의 타이밍 즉 30n가 하나의 로우사이클 타임이었다. 그러나 본 발명에서는 셀블럭어드레스 변환시간으로 3n, 강제프리차지 시간으로 3n가 걸린다면, 한번의 타이밍 15n에 6n가 더해진 21n가 로우사이클 타임이 되는 것이다.

<171> 전술한 실시예에 의한 메모리 장치은 데이터 재저장 시간만큼 데이터 억세스시간이 단축되어 고속 동작이 가능하나, 종래의 메모리 장치에서는 없던 논리적 셀블럭어드레스(LBA)를 물리적 셀블럭어드레스로 변환하는 시간(tBAT) 및 강제 프리차지 시간(tRP)이 로우사이클 타임에 추가되었다.

<172> 본 발명에서는 보다 고속으로 데이터를 억세스하기 위해서 강제 프리차지 시간(tRP)과 셀블럭 어드레스를 변환하는 시간(tBAT)이 줄어들 메모리 장치을 추가로 제안한다.

<173> 도21a은 본 발명의 제2 실시예에 따른 메모리 장치의 블럭 구성도로서, 도21은 도5b에 도시된 한 세그먼트의 명령어 입력부를 나타낸 것이다. 또한 도21에 도시되지 않은 부분은 도5b 및 도6에 도시된 부분과 같다.

<174> 도21a을 참조하여 살펴보면, 제2 실시예에 따른 메모리 장치은 제어부(400)에서 명령어제어부(도1 참조)에서 출력되는 명령어(CMD)를 직접 입력받는 한편, 상기 명령어(CD)를 셀블럭어드레스 변환시간(tBAT) 및 강제 프리차지시간(tRP)만큼 지연시키는 지연부(440)에서 출력되는 지연된 명령어(CMD_Delay)를 입력받는다. 여기서 제1 명령어(CD)로 논리적 셀블럭어드레스를 물리적 셀블럭어드레스로 변환하는 동작과 강제프리차지 동작을 수행하고, 지연된 명령어(CMD_Delay)로는 나머지 동작을 수행하게 된다.

<175> 도21b는 본 발명의 제2 실시예에 따른 개념을 보다 자세히 나타내기 위한 블럭 구성도이다.

<176> 도21b에 도시된 명령어 타이밍제어부(400b)에서는 외부에서 입력되는 명령어를 제1 명령어로 입력받고, 셀블럭어드레스 변환시간(tBAT) 및 강제 프리차지시간(tRP)만큼 지연시키는 지연부(440)에서 출력되는 지연된 명령어(CMD_Delay)를 지연부를 거쳐서 제2 명령어로 입력받아, 제1 명령어(CD)로 논리적 셀블럭어드레스를 물리적 셀블럭어드레스로 변환하는 동작과 셀블럭0,1(500a,500b)에서의 강제 프리차지 동작이 수행되도록 데이터 억세스 제어부(400a)를 제어하고, 제2 명령어는 리드 및 라이트에 필요한 나머지 동작를 셀블럭0,1(500a,500b)이 되도록 데이터 억세스 제어부(400a)를 제어한다.

<177> 도22은 본 발명의 제2 실시예에 따른 메모리 장치의 동작을 나타내는 파형도이다.

<178> 이하에서는 도21a, 도22b 및 도22를 참조하여 제2 실시예에 따른 메모리 장치의 동작을 살펴보면다.

<179> 먼저, 타이밍(t0)에서 제어부(400)는 첫번째 명령을 수행하기 위한 제1 명령어(RD0)를 입력받아, 이 때 입력된 논리적 셀블럭어드레스를 물리적 셀블럭어드레스로 변환하고, 강제 프리차지 동작을 수행한다. 이 때 강제 프리차지동작은 이전 명령어에 대해 억세스되는 셀블럭과 현재의 제1 명령어(RD0)에 대해 억세스될 셀블럭이 같으면, 상기 셀블럭에 재저장없이 강제로 프리차지 시키는 동작을 말한다. 하지만 여기서는 메모리 장치의 첫번째 동작이기 때문에 이전명령이 없기 때문에 실제로는 강제 프리차지 동작은 일어나지 않는다.

<180> 따라서 본 실시예에 따른 메모리 장치은 동작시 명령어를 수행할때 셀블럭어드레스 변환시간(tBAT) 및 강제 프리차지시간(tRP) 만큼의 레이턴시(도22의 'X')를 가지게 되는 것이다.

<181> 이어서, 지연부(440)에서는 제1 명령어(RD0)를 셀블럭어드레스 변환시간(tBAT) 및 강제 프리차지시간(tRP)만큼 지연시킨 제1 명령어(RD0_Delay)를 제어부(400)로 출력한다. 지연된 제1 명령어(RD0)에 의해서 변환된 물리적 셀블럭 어드레스(PBA) 및 로우어드레스(RA)에 해당하는 하나의 워드라인이 활성화되고, 활성화된 워드라인에 해당하는 4K개의 데이터가 대응하는 로컬 비트라인 센스앰프에 의해 감지, 증폭된다. 이어서 증폭된 4K개의 데이터는 글로벌 비트라인 센스앰프로 이동되어 저장된다.

<182> 한편 두번째 명령어(RD1)가 제어부(400)로 입력되어, 함께 입력되는 논리적 셀블럭어드레스(LBA)를 물리적 셀블럭어드레스(PBA)로 변환한다. 이어서 현재 실행중인 제1 명령어(RD0)와 다음에 실행될 제2 명령어(RD1)에 대해서 억세스되는 셀블럭이 같기 때문에 셀블럭0에 대해서 강제 프리차지를 수행한다.(도22의 'Y'구간)

<183> 두번째 타이밍(t1)에서 지연부(440)를 거쳐서 셀블럭어드레스 변환시간(tBAT) 및 강제 프리차지시간(tRP)만큼 지연된 제2 명령어(RD1_D)에 따라 감지 증폭되어 글로벌 비트라인 센스앰프에 저장된 4K개의 데이터중에서 컬럼어드레스에 의해 선택된 데이터는 글로벌 비트라인 센스앰프에서 외부로 출력되고, 예비워드라인이 있는 셀블럭으로 이동되어 재저장동작을 한다. 이 동작에 대한 부분이 도22에 도시된 'intW0'이다.

<184> 또한 이 타이밍(t1)에서 제2 명령어(RD1)에 따라 입력된 어드레스에 의해 하나의 워드라인이 활성화되고, 활성화된 워드라인에 해당하는 4K개의 데이터가 로컬 비트라인 센스앰프에 의해 감지 증폭되어, 글로벌 비트라인센스앰프도 이동된다.

<185> 이어서 세번째 명령어(RD2)가 제어부(400)로 입력되어, 함께 입력되는 논리적 셀블럭어드레스(LBA)를 물리적 셀블럭어드레스(PBA)로 변환한다. 이어서 현재 실행중인 제2 명령어(RD1)와 다음에 실행될 제3 명령어(RD2)에 대해서 억세스되는 셀블럭이 같기 때문에 셀블럭0에 대해서 강제 프리차지를 수행한다.(도22의 'Z'구간)

<186> 다음 타이밍(t2)에서는 지연부(440)를 거쳐서 셀블럭어드레스 변환시간(tBAT) 및 강제 프리차지시간(tRP)만큼 지연된 제3 명령어(RD2_D)에 따라 감지 증폭되어 글로벌 비트라인 센스앰프에 저장된 4K개의 데이터중에서 컬럼어드레스에 의해 선택된 데이터는 글로벌 비트라인 센스앰프에서 외부로 출력되고, 예비워드라인이 있는 셀블럭으로 이동되어 재저장동작을 한다. 이 동작에 대한 부분이 도22에 도시된 'intW1'이다.

<187> 제1 실시예에서는 전술한 바와 같이 전체적인 동작은 제1 실시예에서와 같지만, 명령어를 첫번째로 직접입력받아 해당되는 셀블럭어드레스로 변환하고, 강제프리차지동작을 하고, 두번째로 셀블럭어드레스 변환시간(tBAT) 및 강제 프리차지시간(tRP)만큼 지연시킨 명령어(RD2_D)로 리드 및 라이트에 필요한 동작을 수행하는 것이 다른 점이다.

<188> 이렇게 메모리 장치을 동작시킴으로서, 최초의 동작시에 첫번째 명령어에 대해서는 셀블럭어드레스 변환시간(tBAT) 및 강제 프리차지시간(tRP) 만큼의 레이턴시(도22의 'X')를 가지게 되지만, 두번째 명령어 수행시부터는 로우사이클 타임에서 셀블럭어드레스 변환시간(tBAT) 및 강제 프리차지시간(tRP) 만큼이 줄어들는 효과를 가지고 있다.

<189> 이는 다음명령어에 해당하는 셀블럭어드레스 변환 및 프리차지 동작을 첫번째 타이밍에서 미리 하기 때문이다.

<190> 이를 보다 자세히 살펴보면, 본 발명에 대한 메모리 장치에서는 연속되어 수행되는 명령어중에서 이전의 명령어에 대한 첫번째 타이밍에 증폭된 데이터는 글로벌 비트라인에 저장되어 있어, 이전 명령어에 대해 로컬 비트라인에 저장되어 있던 4K개의 데이터는 강제프리차지 동작에 의해 바로 제거하여도 되며, 이 때 바로 다음 명령어에 대한 셀블럭 어드레스를 변환시키는 것이다.

<191> 이후 다음 명령어가 같은 셀블럭을 억세스하게 되면 글로벌 비트라인에 저장된 데이터는 예비워드라인이 있는 예비셀블럭으로 이동되며, 만약 다음 명령어가 다른 셀블럭을 억세스하게 되면, 다시 원래의 셀블럭에 해당하는 로컬 센스앰프를 이동시키면 되는 것이다. 이 때에서는 한번 증폭된 데이터를 글로벌 센스앰프에서 로컬 센스앰프로 이동시키는 것이므로, 전체적인 로우 사이클 타임에 영향을 미치지는 않는다.

<192> 따라서 제2 실시예에 따른 메모리 장치은 도22에 도시된 바와 같은 로우사이클 타임을 가지게 되며, 수식으로는 제1 실시예에와 비교하여 MAX{0.5*tRP,tINTW}로 정해진다. 이 때 'tRC'는 종래의 로우사이클 타임이고, 'tINTW'는 내부적으로 지정된 예비블럭으로 데이터가 이동되어 재저장하는 시간을 나타낸다. 여기에서는 제1 실시예에에서 제시된 로우사이클 타임에 셀블럭어드레스 변환시간을 나타내는 'tBAT'과, 프리차지 타임을 나타내는 'tRP'가 제거되었다.

<193> 통상 'tINTW'보다는 'tRP'가 더 긴 시간이 걸게 되므로, 제2 실시예에 따른 메모리 장치에서는 종래대비 최대 1/2까지 로우사이클 타임을 줄이는 것이 가능하다.

<194> 본 발명의 기술 사상은 상기 바람직한 실시예에 따라 구체적으로 기술되었으나, 상기한 실시예는 그 설명을 위한 것이며 그 제한을 위한 것이 아님을 주의하여야 한다. 또

한, 본 발명의 기술 분야의 통상의 전문가라면 본 발명의 기술 사상의 범위내에서 다양한 실시예가 가능함을 이해할 수 있을 것이다.

【발명의 효과】

<195> 본 발명의 의한 메모리 장치은 동일 셀블럭에 연속적인 데이터가 억세스되던지, 또는 서로 다른 셀블럭 간에 교대로 데이터를 억세스하든지, 데이터 억세스 패턴에 상관없이 항상 고속으로 데이터를 연속해서 억세스할 수 있기 때문에; 본 발명의 메모리 장치이 메인메모리로 구비된 시스템은 데이터 억세스 패턴에 상관없이 전체적인 시스템 속도가 향상되는 효과를 기대할 수 있다.

<196> 또한 본 발명에 의한 메모리 장치은 종래의 메모리 장치 구조를 최대한으로 유지시키면서 태그관련 블럭 몇개만을 추가함으로써, 고속 데이터 억세스를 구현했기 때문에 제조개발 비용을 최대한 줄이면서도 고속으로 동작하는 메모리 장치을 제조할 수 있다.

【특허청구범위】

【청구항 1】

다수개의 단위셀을 구비하는 제1 셀블럭;

상기 제1 셀블럭에 억세스되는 데이터 백업을 위한 제2 셀블럭; 및

상기 제1 셀블럭에 대해 연속적으로 제1 데이터 및 제2 데이터가 억세스될 때, 상기 제1 셀블럭에서는 상기 제1 데이터의 재저장동작을 수행하지 않고 상기 제2 데이터가 억세스되도록 제어하고, 상기 제2 셀블럭에서는 상기 제1 데이터의 재저장 동작이 수행하도록 제어하기 위한 제어부

를 구비하는 메모리 장치.

【청구항 2】

제 1 항에 있어서,

상기 제1 데이터를 상기 제1 셀블럭으로부터 전달받아 래치하기 위한 래치 수단; 및

상기 래치수단에 래치된 데이터를 상기 제2 셀블럭으로 전달시키기 위한 신호라인을 더 구비하는 것을 특징으로 하는 메모리 장치.

【청구항 3】

각각 M개의 워드라인을 가지고 입력되는 로우어드레스에 대응하도록 N개로 구성된 단위블럭에, 추가적으로 상기 M개의 워드라인을 가지는 단위블럭을 더 포함하여 N+1개의 단위블럭으로 구성된 셀블럭;

상기 N+1개의 단위셀블럭중에서 선택된 하나의 단위 셀블럭으로 부터 억세되는 데이터를 임의의 단위 블럭 또는 다른 단위셀블럭으로 재저장시키도록 제어하는 제어수단;

제 3 항에 있어서,

상기 제어수단은

입력되는 로우 어드레스에 대응하여 2개의 단위블럭이 활성하 되도록 제어하는 제어수단을 구비하는

【청구항 4】

제 3 항에 있어서,

상기 제어수단은

상기 N+1개의 셀블럭에 대한 상기 논리적 셀블럭 어드레스를 저장하기 위한 태그테이블;

상기 M개의 워드라인에 대한 M개의 예비블럭 정보를 저장하기 위한 예비블럭 테이블; 및

상기 테그테이블 및 예비블럭테이블을 제어하기 위한 태그제어부를 구비하는 것을 특징으로 하는 메모리 장치.

【청구항 5】

제 4 항에 있어서,

상기 테그제어부는

논리적 셀블럭 어드레스에 대응되는 셀블럭에 상기 제1 및 제2 데이터가 억세스될 수 있도록 상기 N+1개의 셀블럭 및 상기 태그테이블을 제어하고, 상기 M개의 워드라인중 선택된 하나의 워드라인에 대응하는 예비블럭을 상기 N+1개의 셀블럭 중에서 찾기 위해 상기 예비블럭을 제어하는 것을 특징으로 하는 메모리 장치.

【청구항 6】

입력되는 어드레스에 대응하는 셀블럭이 N개인 메모리 장치에 있어서,

활성화된 하나의 워드라인에 대응되는 K개의 비트라인에 인가되는 K개의 데이터를 감지증폭하기 위한 로컬비트라인 센스앰프부를 각각 구비한 N+1개의 셀블럭;

상기 로컬비트라인 센스앰프부에 의해 증폭된 K개의 데이터를 래치하기 위한 글로벌 비트라인 센스앰프;

N+1 개의 셀블럭에 각각 구비된 로컬비트라인 센스앰프부와 상기 글로벌 비트라인 센스앰프를 연결하기 위한 글로벌 비트라인; 및

상기 N+1개의 셀블럭중 선택된 하나의 셀블럭에 제1 및 제2 워드라인이 연속적으로 활성화될 때, 상기 제1 워드라인에 대해 응답하여 감지증폭된 K개의 데이터는 상기 글로벌 비트라인 센스앰프 및 상기 글로벌 비트라인을 통해 상기 제1 워드라인에 대한 예비 블럭으로 이동시켜 재저장시키고, 상기 제2 워드라인에 대한 K개의 데이터를 감지, 증폭되도록 재어하는 제어수단

을 구비하는 것을 특징으로 하는 메모리 장치.

【청구항 7】

제 6 항에 있어서,

상기 글로벌 비트라인을 N+1개의 셀블럭에 각각 구비된 로컬비트라인 센스앰프부과 선택적으로 연결하기 위한 글로벌 비트라인 연결부를 더 구비한 것을 특징으로 하는 메모리 장치.

【청구항 8】

입력되는 어드레스에 대응하는 셀블럭이 N개인 메모리 장치에 있어서, M개의 워드라인을 각각 구비하며, 예비 워드라인을 상기 M개 만큼 더 구비하기 위해 배치된 N+1개의 셀블럭을 구비하는 메모리 장치의 구동방법에 있어서,

상기 N+1개의 셀블럭 중에서 하나인 제1 셀블럭을 선택하는 제1 단계;

상기 제1 셀블럭에 구비된 제1 워드라인을 활성화시키는 제2 단계;

상기 제1 워드라인에 대응되는 K개의 데이터를 감지 증폭하는 제3 단계;

감지증폭된 상기 제1 워드라인에 대응되는 K개의 데이터를 상기 제1 워드라인에 대응하는 예비워드라인이 구비된 셀블럭으로 이동시켜 재저장하는 제4 단계;

상기 제1 셀블럭에 제2 워드라인을 활성화시키는 제5 단계; 및

상기 제2 워드라인에 대응하는 K개의 데이터를 감지증폭하는 제6 단계를 포함하며, 제4 내지 제6 단계의 수행은 동시에 이루어진것을 특징으로 하는 메모리 장치의 구동방법.

【청구항 9】

제 8 항에 있어서,

상기 제1 워드라인에 해당되는 예비워드라인이 구비된 셀블럭으로 이동시켜 재저장하는 단계는

상기 제1 워드라인에 대응하는 K개의 데이터를 글로벌 비트라인을 통해 글로벌 비트라인 센스앰프에 래치시키는 단계;

상기 글로벌 비트라인 센스앰프에 래치된 K개의 데이터를 상기 예비워드라인이 구비된 셀블럭으로 이동시키여 재저장하는 단계; 및

상기 제1 셀블럭에서 증폭된 제1 워드라인에 대응하는 K개의 데이터를 강제 프리차지 시켜 제거하는 단계를 포함하여 이루어지는 것을 특징으로 하는 메모리 장치의 구동방법.

【청구항 10】

입력되는 어드레스에 대응하는 셀블럭이 N개인 메모리 장치에 있어서,

M 개의 워드라인을 각각 구비하며, 예비 워드라인을 상기 M개 만큼 더 구비하기 위해 배치된 N+1개의 셀블럭;

N개의 셀블럭을 선택하기 위한 논리적 어드레스를 입력받아 N+1개의 셀블럭중 하나를 선택하기 위한 물리적 어드레스로 변환하여 출력하기 위한 어드레스 변환수단;

상기 N+1개의 셀블럭중에서 하나인 제1 셀블럭에 제1 및 제2 데이터가 연속적으로 데이터가 억세스될 때에 제1 데이터를 래치하기 위한 래치수단;

상기 물리적 어드레스를 입력받아 상기 N+1개의 셀블럭에 데이터를 억세스 하며, 상기 래치수단에 데이터가 래치되면 상기 제1 셀블럭에 억세스된 제1 데이터를 강제프리차지시키는 데이터억세스 제어부;

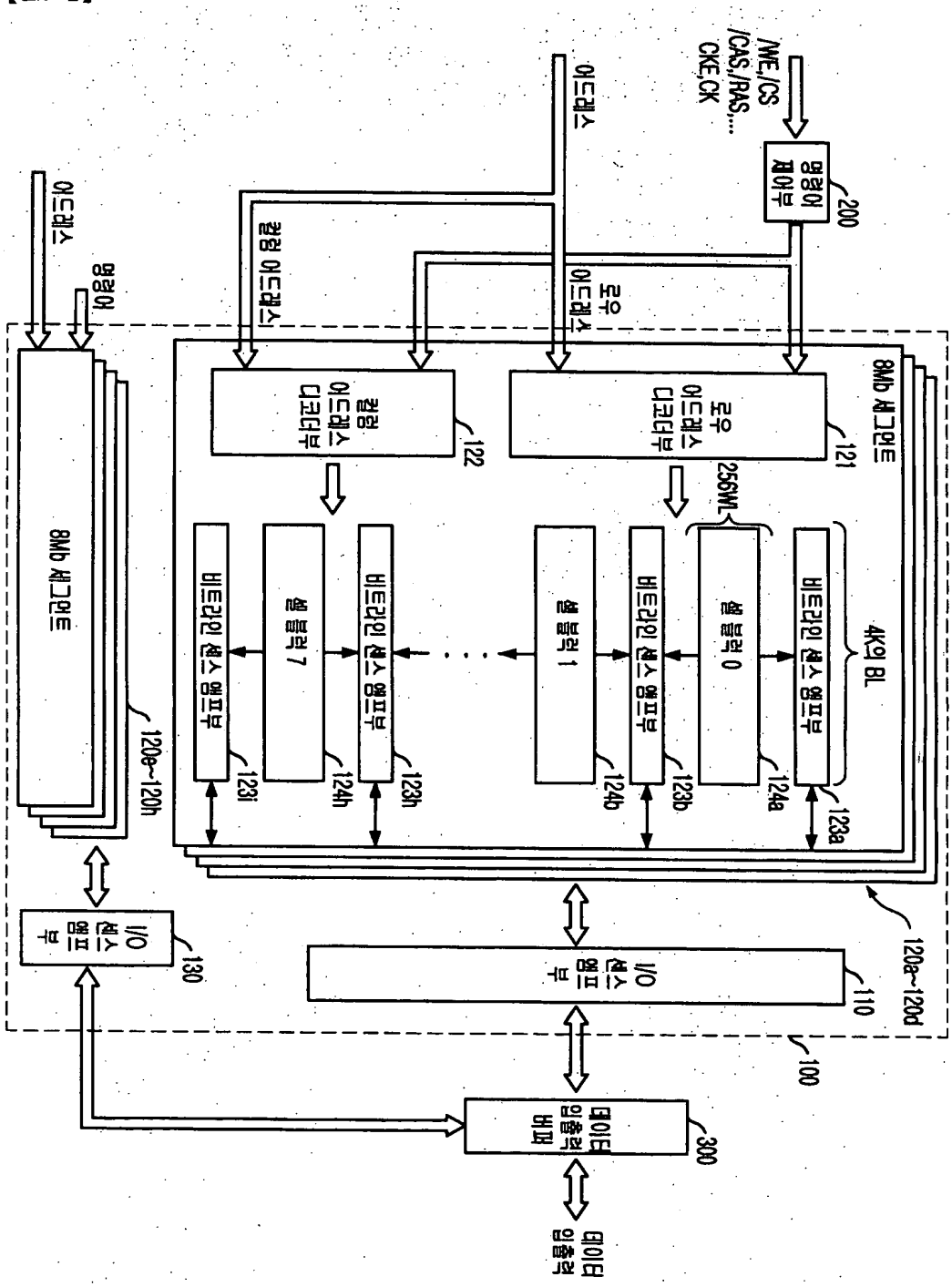
데이터 억세스를 위한 제1 명령어를 입력받아 상기 어드레스 변환시간 및 상기 강제프리차지 타임만큼 지연시켜 제2 명령어를 출력하기 위한 지연수단; 및

상기 제2 명령어를 입력받아 상기 데이터억세스 제어부의 강제프리차지동작 및 상기 어드레스 변환수단을 제어하고, 상기 제1 명령어를 입력받아 상기 셀블럭에 데이터가 억세스되도록 상기 데이터 억세스 제어부를 제어하는 타이밍 제어부

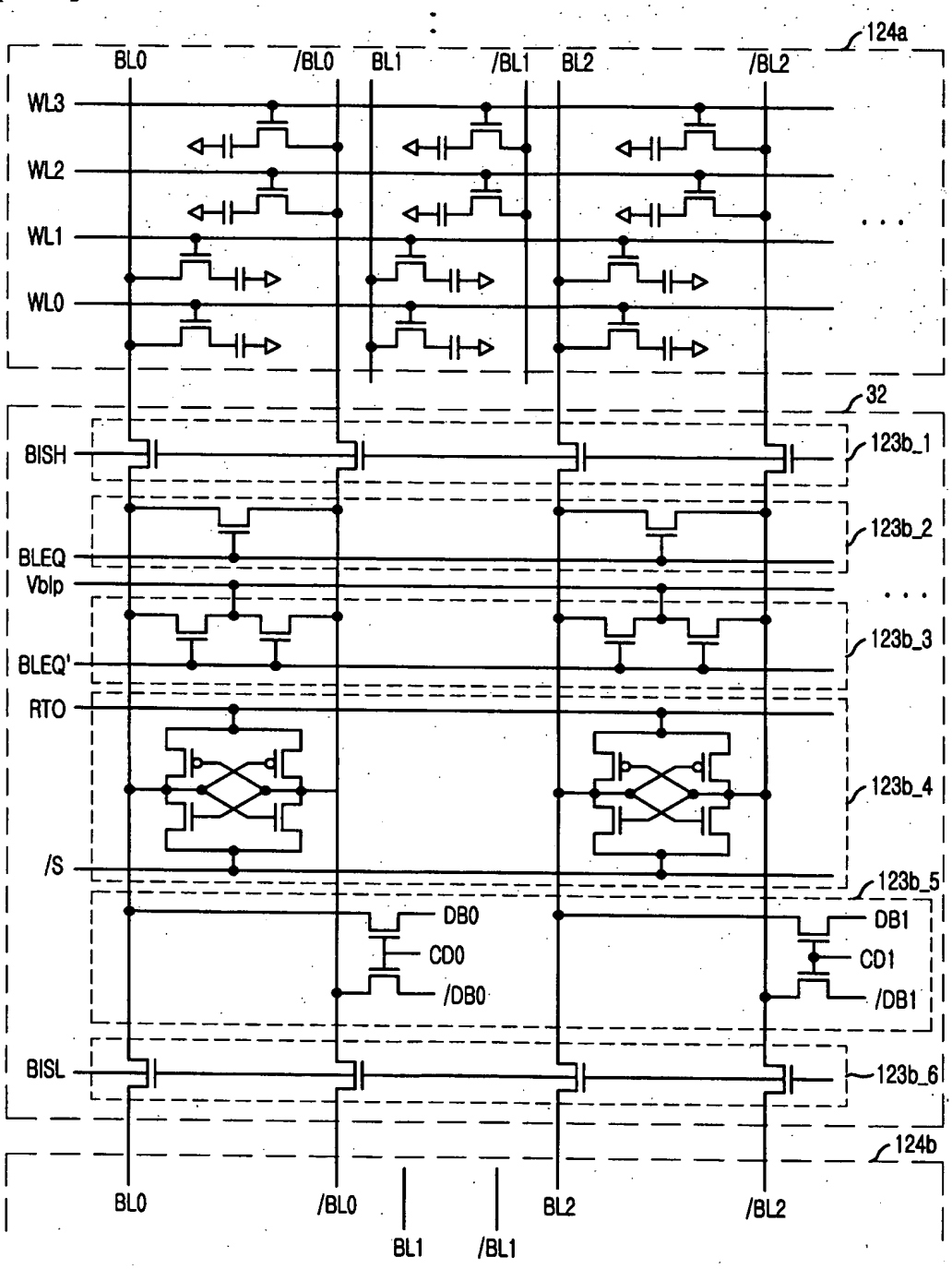
를 구비하는 메모리 장치.

【도면】

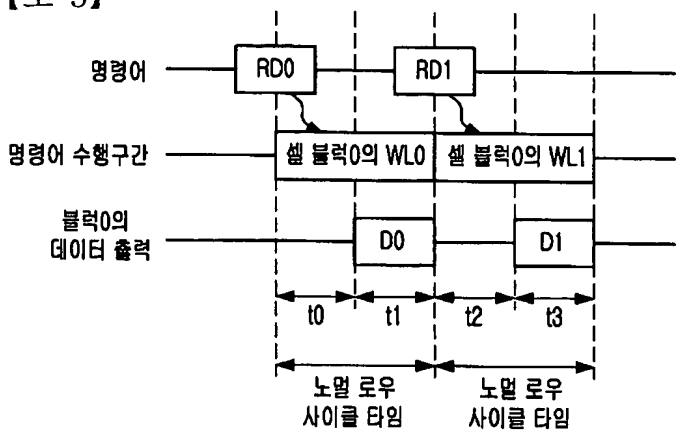
【도 1】


/WE,/CS
/CAS,/RAS,...
CKE,CK
명령어 제어부
200
어드레스
컬럼 어드레스
로우 어드레스
8Mb 세그먼트
컬럼 어드레스 디코더부
122
로우 어드레스 디코더부
121
256WL
4K의 BL
비트라인 센스 앰프부
123a
123b
123h
123i
셀 블럭 0
셀 블럭 1
셀 블럭 7
124a
124b
124h
8Mb 세그먼트
120e~120h
120a~120d
I/O 센스 앰프부
130
110
100
데이타 입출력 버퍼
300
데이타 입출력

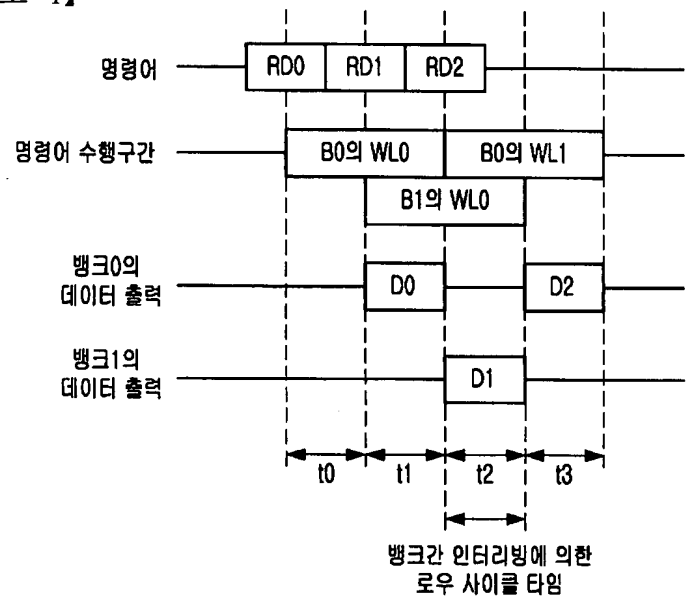
【도 2】


124a
BL0
/BL0
BL1
/BL1
BL2
/BL2
WL3
WL2
WL1
WL0
32
BISH
123b_1
123b_2
BLEQ
Vblp
123b_3
BLEQ'
RTO
123b_4
/S
123b_5
DB0
CD0
/DB0
DB1
CD1
/DB1
BISL
123b_6
124b
BL0
/BL0
BL1
/BL1
BL2
/BL2

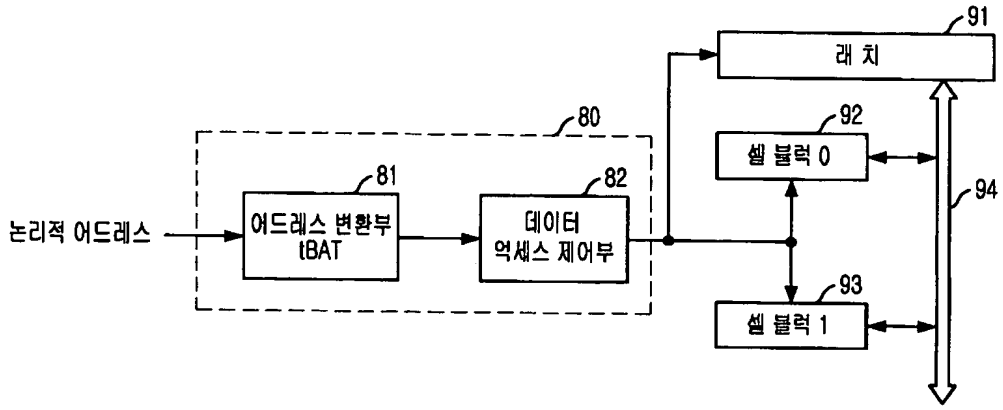
【도 3】


명령어
RD0
RD1
명령어 수행구간
셀 블럭0의 WL0
셀 블럭0의 WL1
블럭0의
데이터 출력
D0
D1
t0
t1
t2
t3
노멀 로우
사이클 타임
노멀 로우
사이클 타임

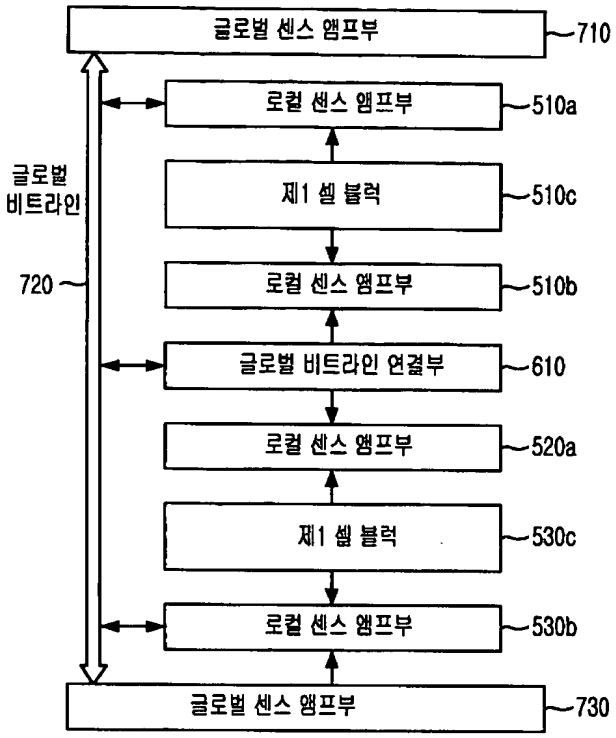
【도 4】


명령어
RD0
RD1
RD2
명령어 수행구간
B0의 WL0
B0의 WL1
B1의 WL0
뱅크0의
데이터 출력
D0
D2
뱅크1의
데이터 출력
D1
t0
t1
t2
t3
뱅크간 인터리빙에 의한
로우 사이클 타임

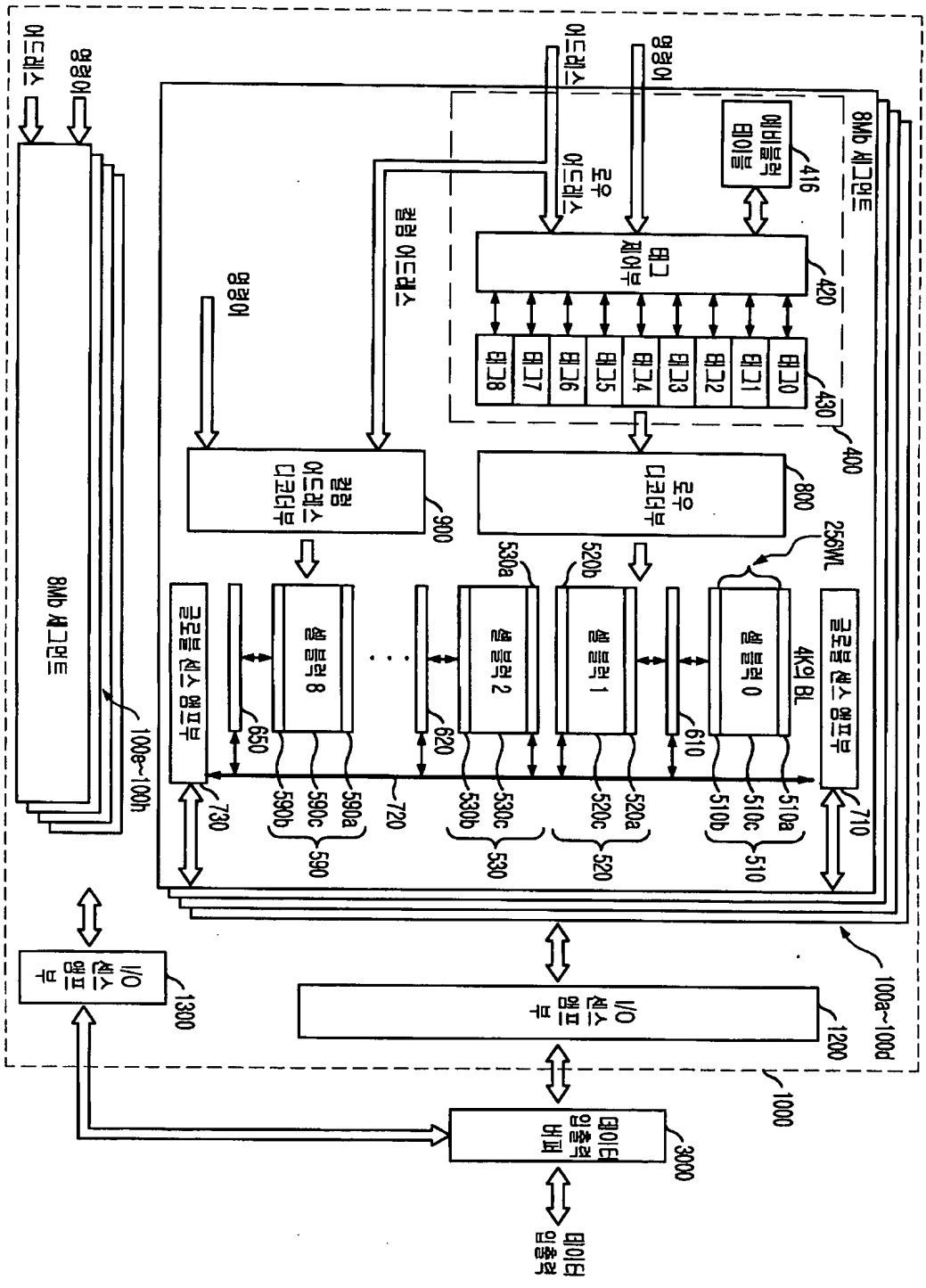
【도 5a】


91
래 치
80
92
셀 블럭 0
81
82
94
논리적 어드레스
어드레스 변환부
tBAT
데이터
억세스 제어부
93
셀 블럭 1

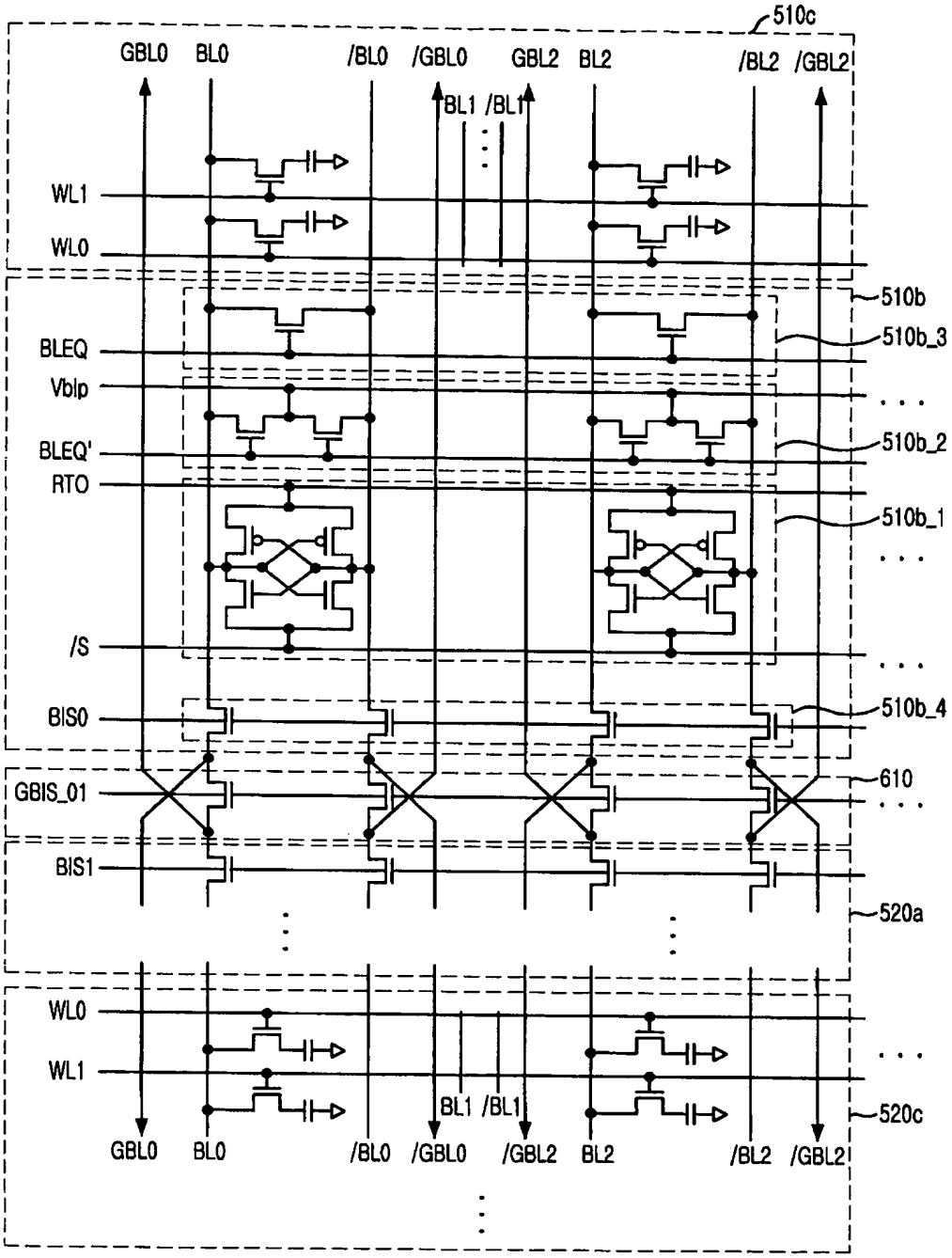
【도 5b】


글로벌 센스 앰프부
710
로컬 센스 앰프부
510a
글로벌
비트라인
제1 셀 블럭
510c
720
로컬 센스 앰프부
510b
글로벌 비트라인 연결부
610
로컬 센스 앰프부
520a
제1 셀 블럭
530c
로컬 센스 앰프부
530b
글로벌 센스 앰프부
730

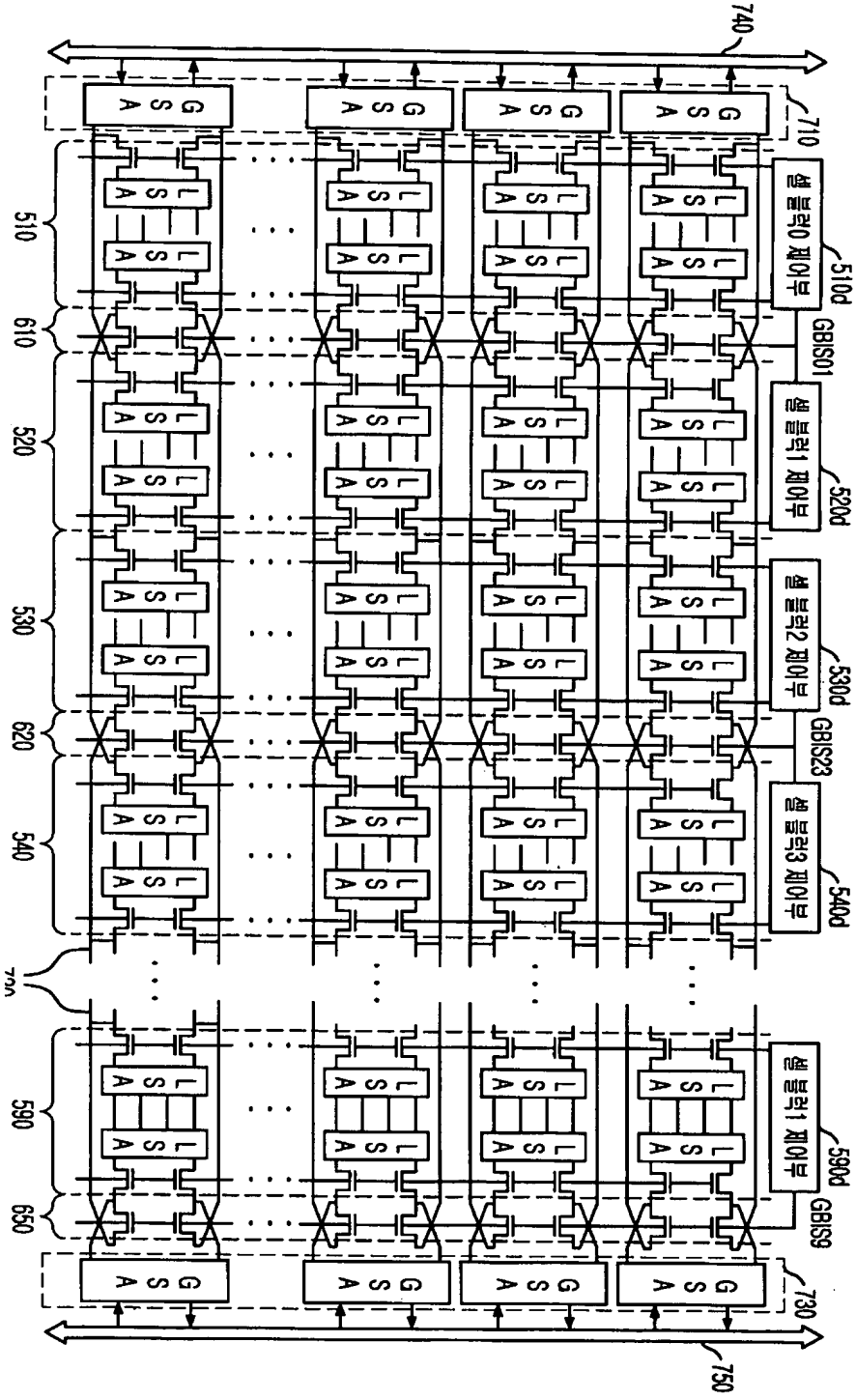
【도 5c】


어드레스
명령어
8Mb 세그먼트
100e~100h
I/O 센스 앰프부
1300
I/O 센스 앰프부
1200
100a~100d
1000
데이터 입출력 버퍼
3000
데이터 입출력
어드레스
명령어
로우 어드레스
컬럼 어드레스
리던던시 테이블
416
태그 제어부
420
태그0
태그1
태그2
태그3
태그4
태그5
태그6
태그7
태그8
430
400
컬럼 어드레스 디코더부
900
로우 디코더부
800
256WL
4K의 BL
글로벌 센스 앰프부
710
셀 블럭 0
셀 블럭 1
셀 블럭 2
셀 블럭 8
510a
510b
510c
510
520a
520b
520c
520
530a
530b
530c
530
590a
590b
590c
590
610
620
650
720
730

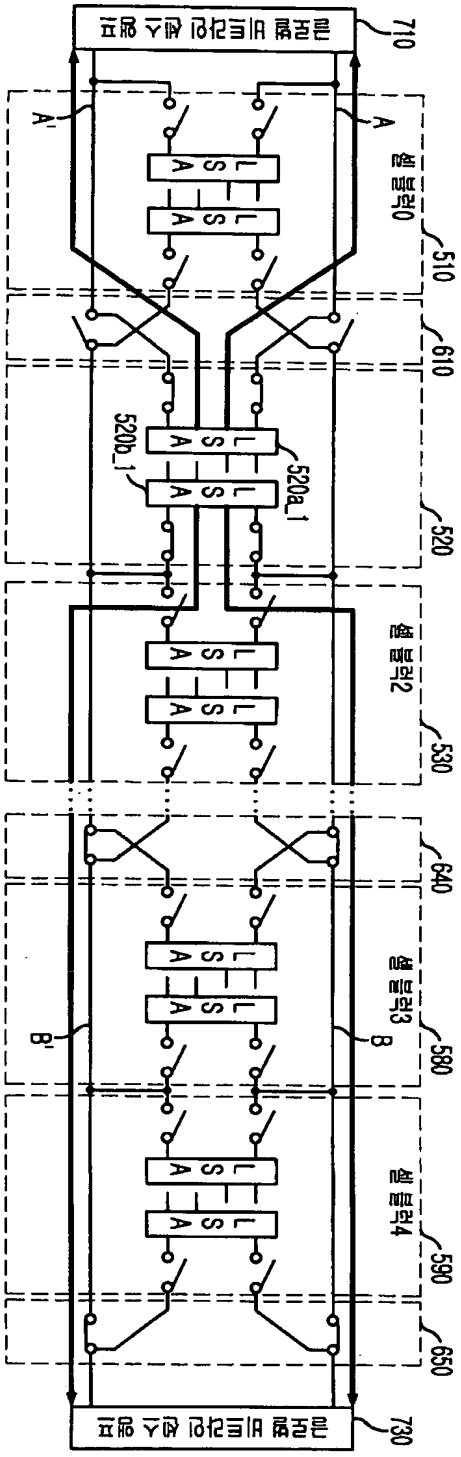
【도 6】


510c
GBL0
BL0
/BL0
/GBL0
GBL2
BL2
/BL2
/GBL2
BL1
/BL1
WL1
WL0
510b
510b_3
BLEQ
Vblp
510b_2
BLEQ'
RTO
510b_1
/S
510b_4
BIS0
610
GBIS_01
BIS1
520a
WL0
WL1
520c
GBL0
BL0
/BL0
/GBL0
/GBL2
BL2
/BL2
/GBL2
BL1
/BL1

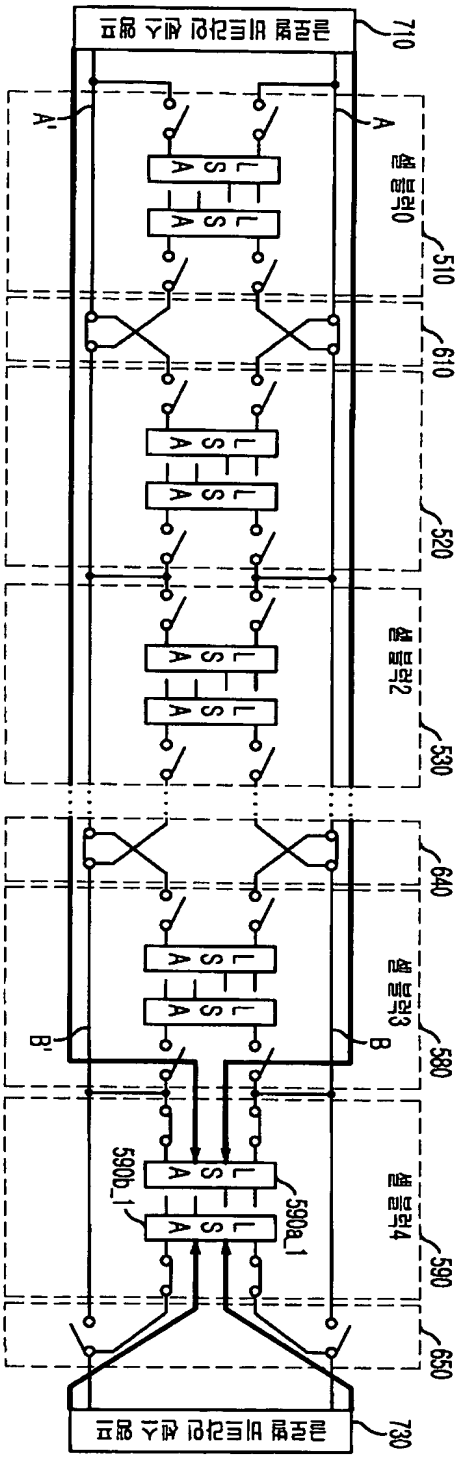
【도 7】


740
710
510
610
520
530
620
540
590
650
730
750
510d
520d
530d
540d
590d
GBIS01
GBIS23
GBIS9
GSA
LSA
SA
셀블럭0 제어부
셀블럭1 제어부
셀블럭2 제어부
셀블럭3 제어부
셀블럭1 제어부

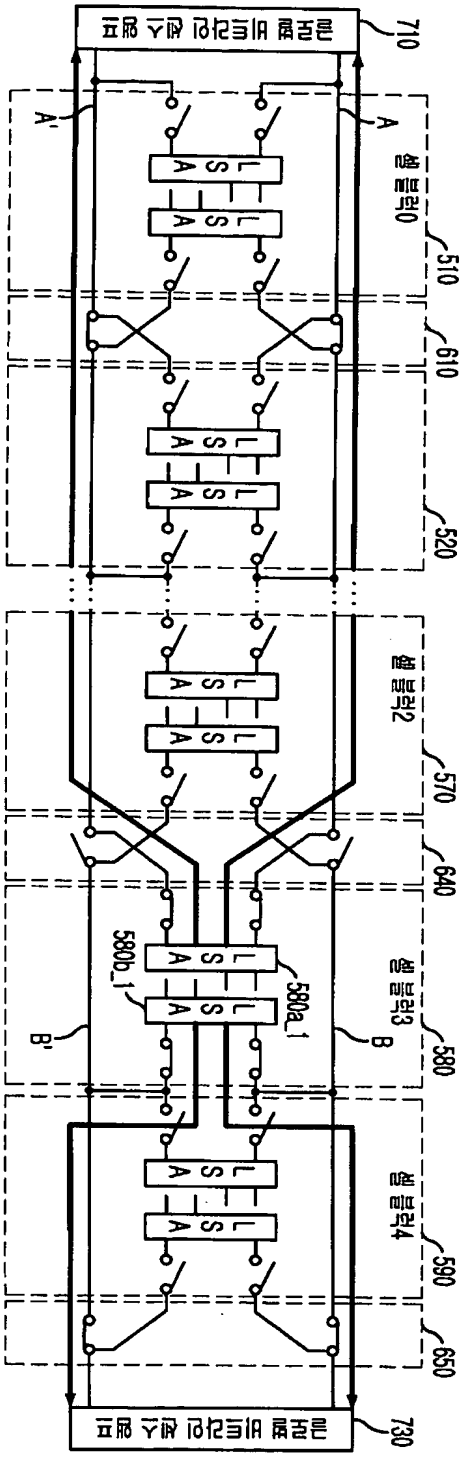
【도 8a】


글로벌 버티컬 센스 앰프
710
A'
A
LSA
LSA
셀 블럭0
510
610
520b_1
520a_1
520
셀 블럭2
530
640
셀 블럭3
B'
B
580
셀 블럭4
590
650
글로벌 버티컬 센스 앰프
730

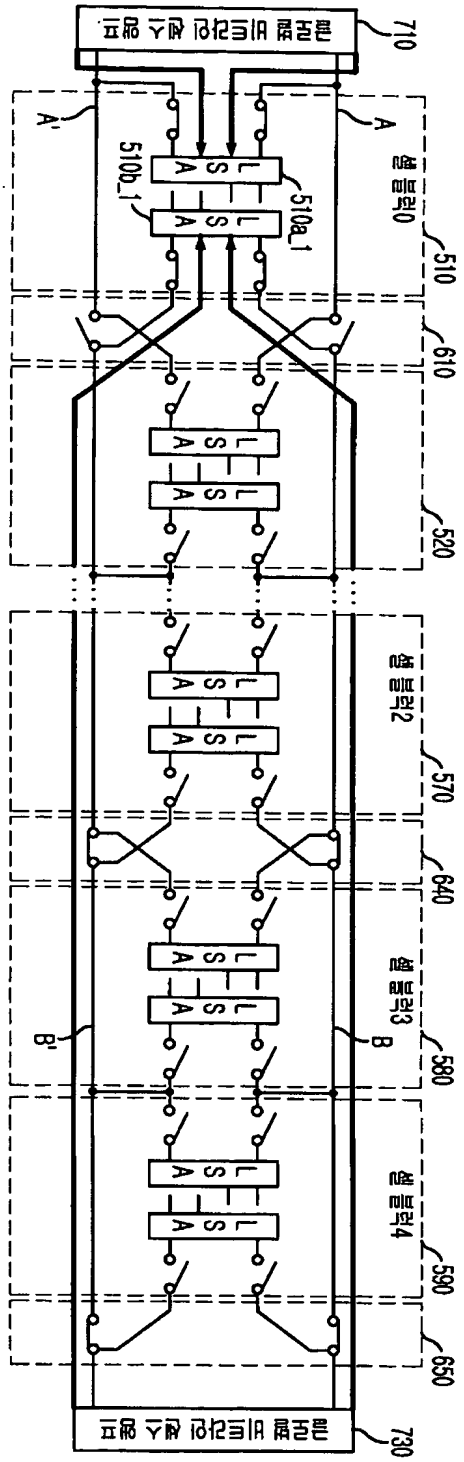
【도 8b】


글로벌 버티칼 센스 앰프
710
A'
A
SA
LS
서브블럭0
510
610
520
서브블럭2
530
640
서브블럭3
B'
B
580
590b_1
590a_1
서브블럭4
590
650
글로벌 버티칼 센스 앰프
730

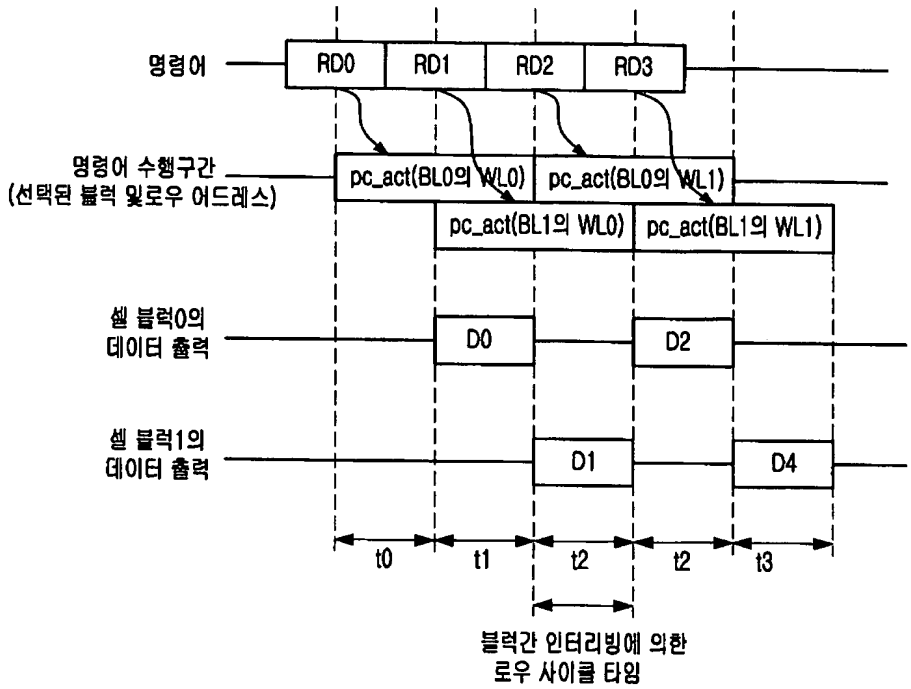
【도 8c】


글로벌 버티컬 센스 앰프
710
A'
A
메모리 블록0
510
610
520
메모리 블록2
570
640
580b_1
580a_1
B'
B
메모리 블록3
580
메모리 블록4
590
650
글로벌 버티컬 센스 앰프
730

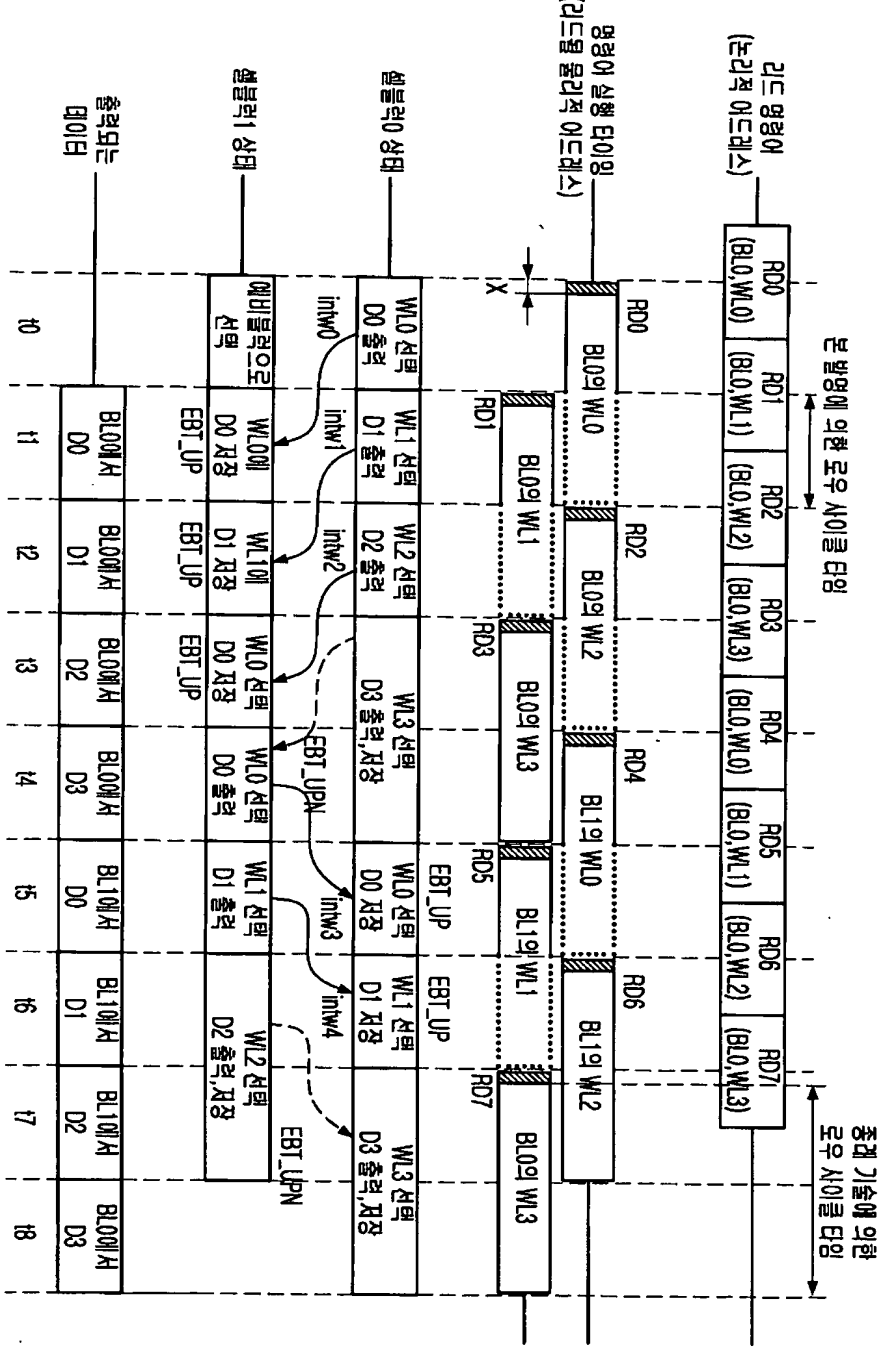
【도 8d】


글로벌 버티컬 센스 앰프
710
A'
A
510b_1
510a_1
A S L
섹션블럭0
510
610
520
섹션블럭2
570
640
섹션블럭3
B'
B
580
섹션블럭4
590
650
글로벌 버티컬 센스 앰프
730

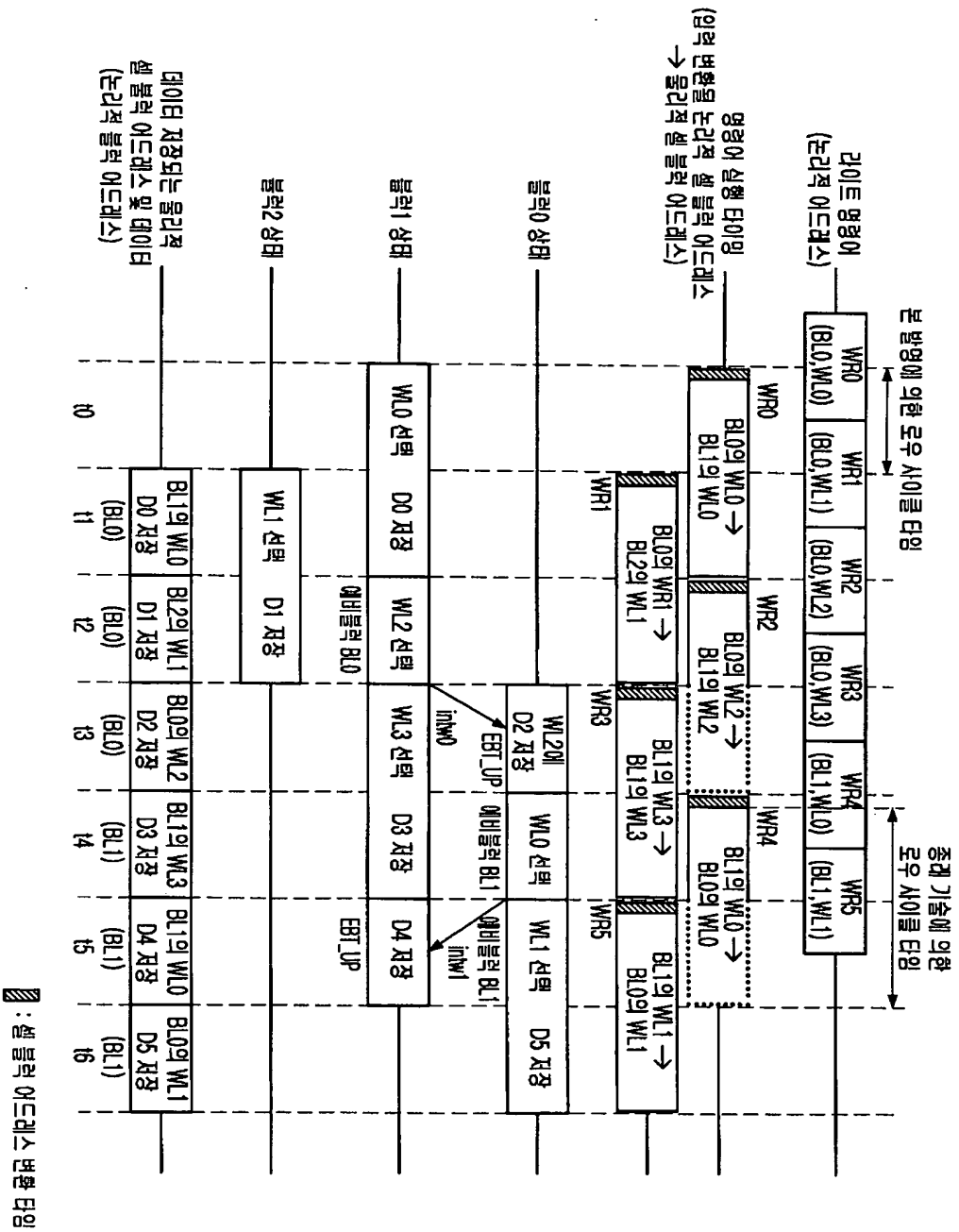
【도 9】


명령어
RD0
RD1
RD2
RD3
명령어 수행구간
(선택된 뱅크 및로우 어드레스)
pc_act(BL0의 WL0)
pc_act(BL0의 WL1)
pc_act(BL1의 WL0)
pc_act(BL1의 WL1)
셀 뱅크0의
데이터 출력
D0
D2
셀 뱅크1의
데이터 출력
D1
D4
t0
t1
t2
t2
t3
뱅크간 인터리빙에 의한
로우 사이클 타임

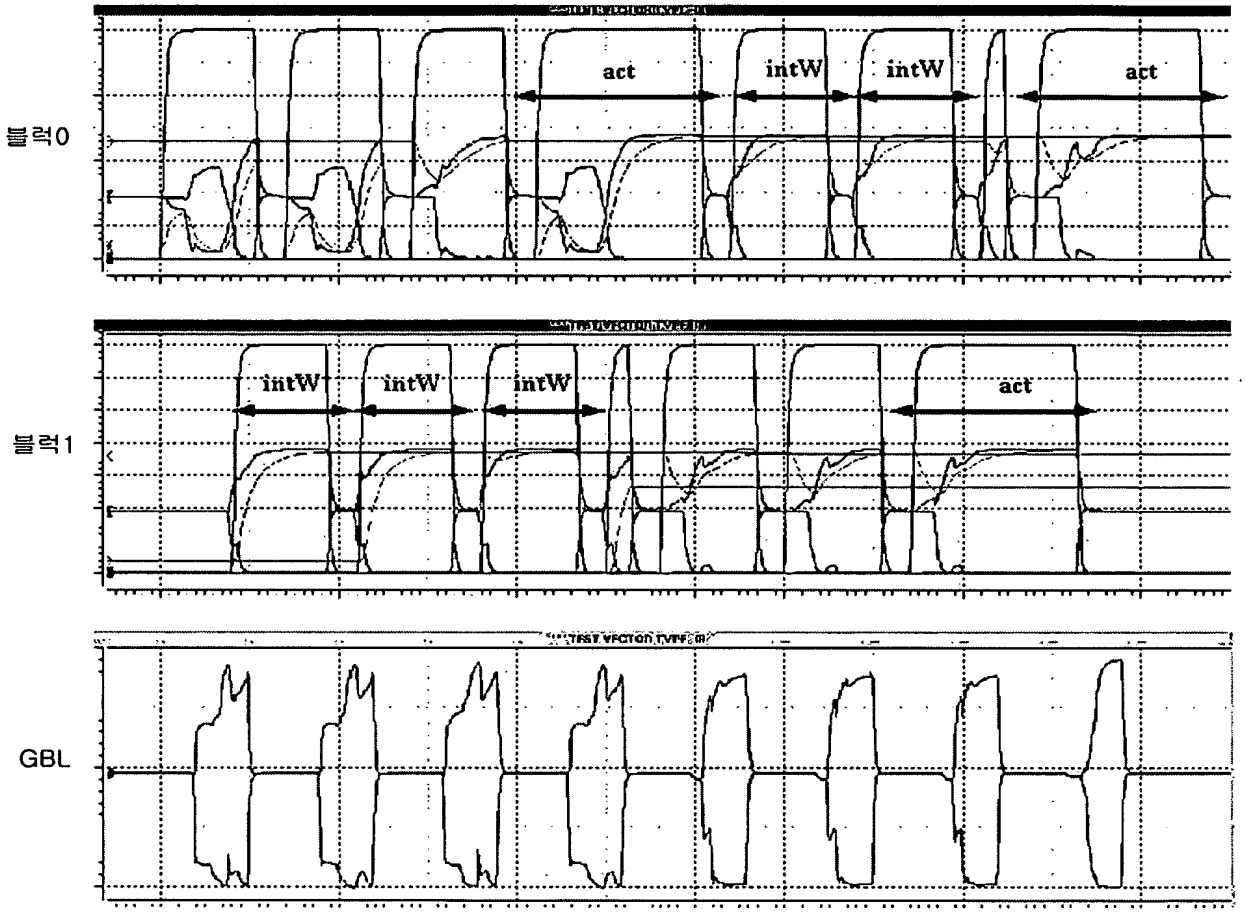
【도 10】


리드 명령어 (논리적 어드레스)
명령어 실행 타이밍 (리드할 물리적 어드레스)
본 발명에 의한 로우 사이클 타임
종래 기술에 의한 로우 사이클 타임
셀블럭0 상태
셀블럭1 상태
출력되는 데이터
RD0 (BL0,WL0)
RD1 (BL0,WL1)
RD2 (BL0,WL2)
RD3 (BL0,WL3)
RD4 (BL0,WL0)
RD5 (BL0,WL1)
RD6 (BL0,WL2)
RD7 (BL0,WL3)
BL0의 WL0
BL0의 WL1
BL0의 WL2
BL0의 WL3
BL1의 WL0
BL1의 WL1
BL1의 WL2
BL0의 WL3
X
WL0 선택 D0 출력
WL1 선택 D1 출력
WL2 선택 D2 출력
WL3 선택 D3 출력,저장
WL0 선택 D0 저장
WL1 선택 D1 저장
WL2 선택 D2 출력,저장
WL3 선택 D3 출력,저장
예비블럭으로 선택
WL0에 D0 저장
WL1에 D1 저장
WL0 선택 D0 저장
WL1 선택 D1 출력
WL2 선택 D2 출력,저장
intw0
intw1
intw2
EBT_UP
EBT_UPN
intw3
intw4
BL0에서 D0
BL0에서 D1
BL0에서 D2
BL0에서 D3
BL1에서 D0
BL1에서 D1
BL1에서 D2
BL0에서 D3
t0
t1
t2
t3
t4
t5
t6
t7
t8
: 셀 블럭 어드레스 변환 타임

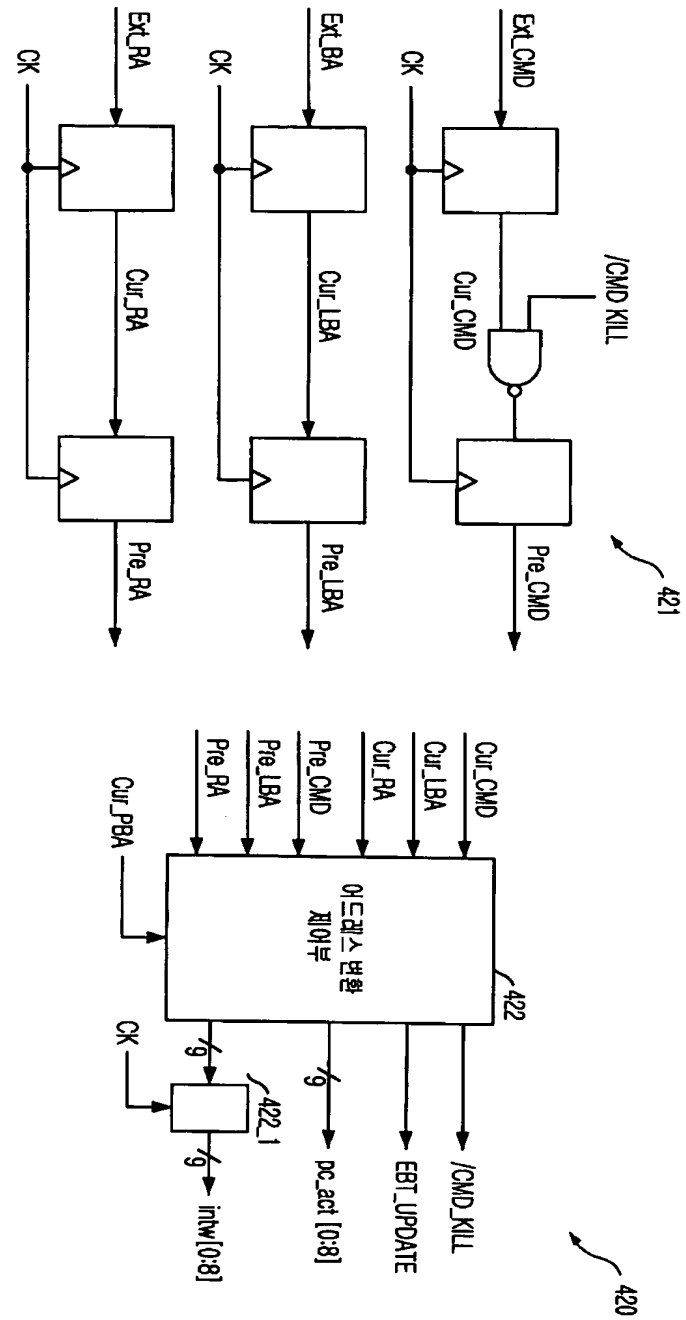
【도 11】


라이트 명령어 (논리적 어드레스)
명령어 실행 타이밍 (입력 변환된 논리적 셀 블럭 어드레스 → 물리적 셀 블럭 어드레스)
블럭0 상태
블럭1 상태
블럭2 상태
데이터 저장되는 물리적 셀 블럭 어드레스 및 데이터 (논리적 블럭 어드레스)
WR0 (BL0,WL0)
WR1 (BL0,WL1)
WR2 (BL0,WL2)
WR3 (BL0,WL3)
WR4 (BL1,WL0)
WR5 (BL1,WL1)
BL0의 WL0 → BL1의 WL0
BL0의 WR1 → BL2의 WL1
BL0의 WL2 → BL1의 WL2
BL1의 WL3 → BL1의 WL3
BL1의 WL0 → BL0의 WL0
BL1의 WL1 → BL0의 WL1
WL0 선택 D0 저장
WL2 선택 D2 저장
WL3 선택 D3 저장
D4 저장
WL1 선택 D1 저장
WL2에 D2 저장
WL0 선택 D5 저장
예비블럭 BL0
예비블럭 BL1
EBT_UP
intw0
intw1
BL1의 WL0 D0 저장
BL2의 WL1 D1 저장
BL0의 WL2 D2 저장
BL1의 WL3 D3 저장
BL1의 WL0 D4 저장
BL0의 WL1 D5 저장
(BL0)
(BL1)
t0
t1
t2
t3
t4
t5
t6
동일 명령에 의한 로우 사이클 타임
종래 기술에 의한 로우 사이클 타임
: 셀 블럭 어드레스 변환 타임

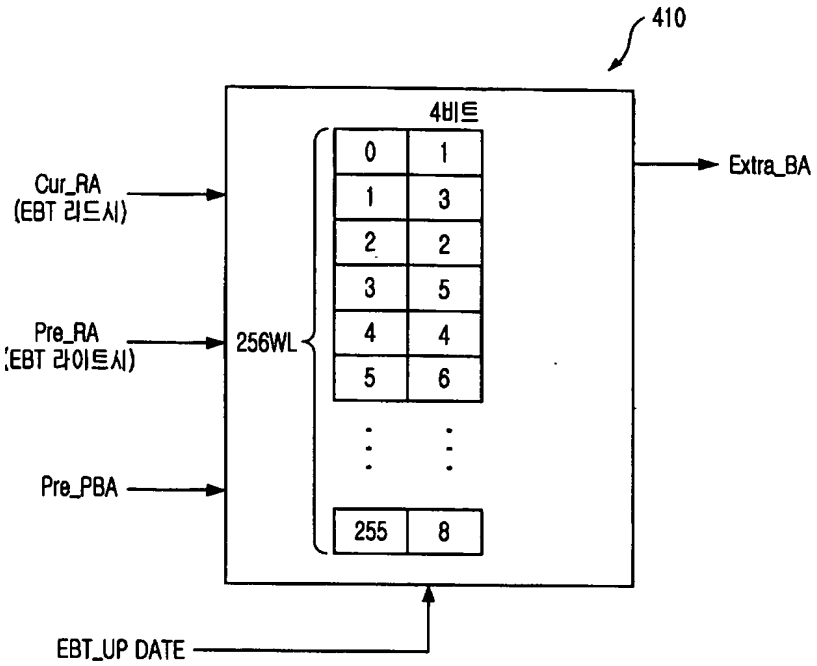
【도 12】



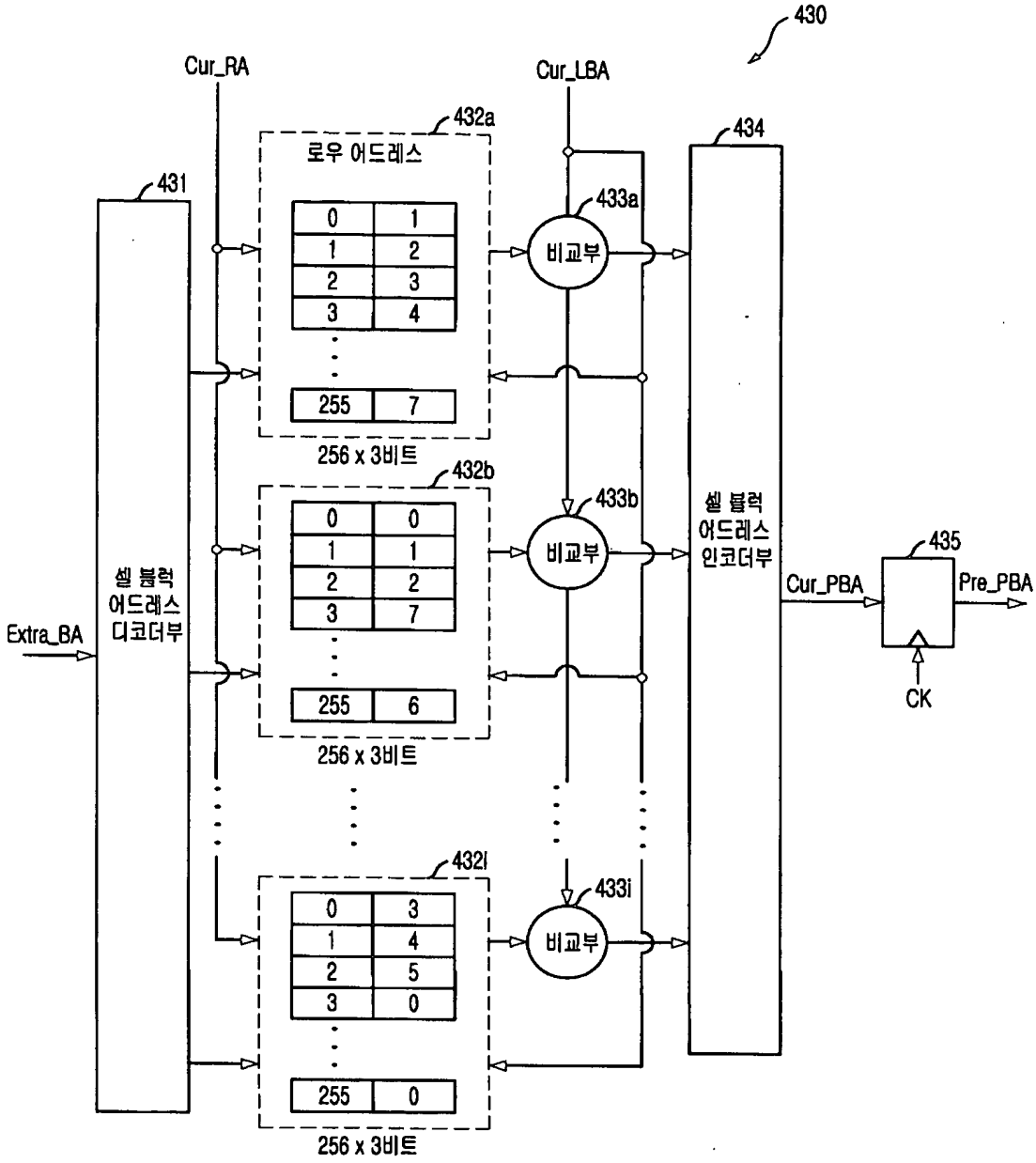
【도 13】


Ext_CMD
CK
Cur_CMD
/CMD KILL
Pre_CMD
421
Ext_BA
Cur_LBA
Pre_LBA
Ext_RA
Cur_RA
Pre_RA
Cur_CMD
Cur_LBA
Cur_RA
Pre_CMD
Pre_LBA
Pre_RA
Cur_PBA
어드레스 변환 제어부
422
9
422_1
pc_act [0:8]
EBT_UPDATE
/CMD_KILL
intw[0:8]
420

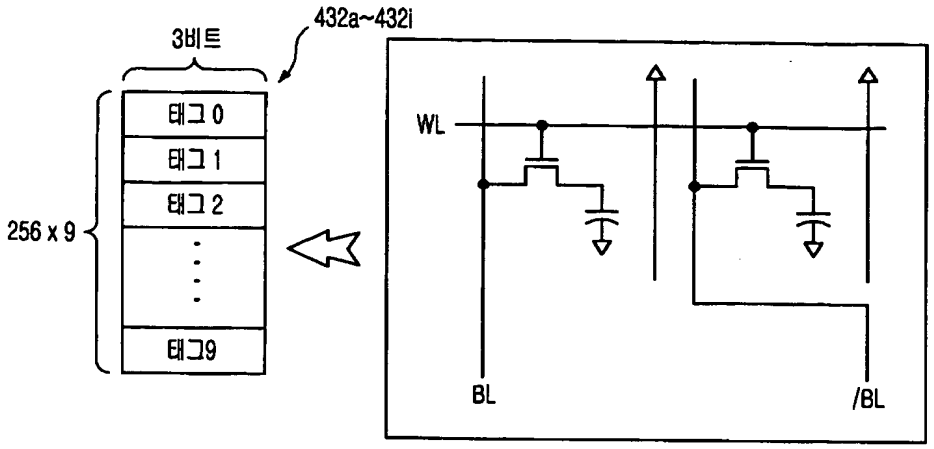
【도 14】


410
4비트
Cur_RA
(EBT 리드시)
Pre_RA
(EBT 라이트시)
Pre_PBA
256WL
0 1
1 3
2 2
3 5
4 4
5 6
255 8
Extra_BA
EBT_UP DATE

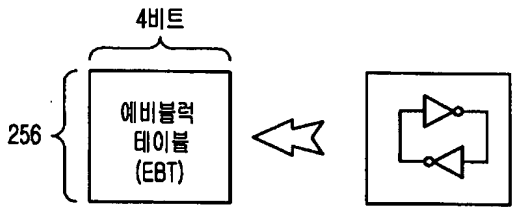
【도 15】


430
Cur_RA
Cur_LBA
432a
로우 어드레스
434
431
433a
비교부
0 1
1 2
2 3
3 4
255 7
256 x 3비트
432b
433b
비교부
셀 블럭
어드레스
인코더부
435
Cur_PBA
Pre_PBA
Extra_BA
셀 블럭
어드레스
디코더부
0 0
1 1
2 2
3 7
255 6
256 x 3비트
CK
432l
433i
비교부
0 3
1 4
2 5
3 0
255 0
256 x 3비트

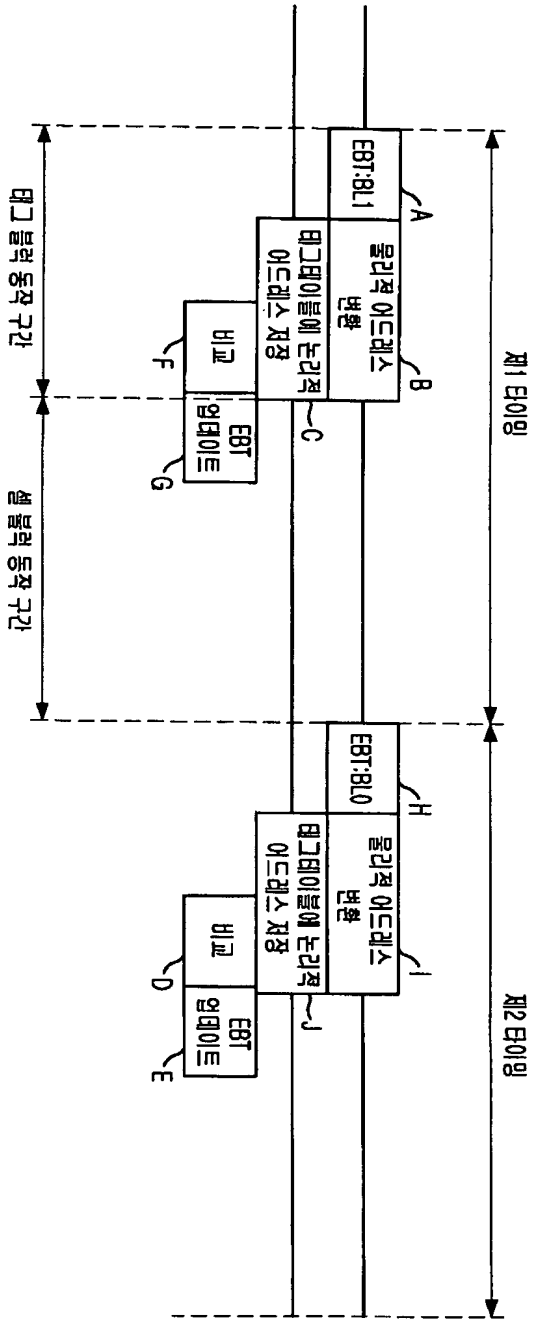
【도 16】


3비트
432a~432i
태그 0
태그 1
태그 2
태그9
256 x 9
WL
BL
/BL

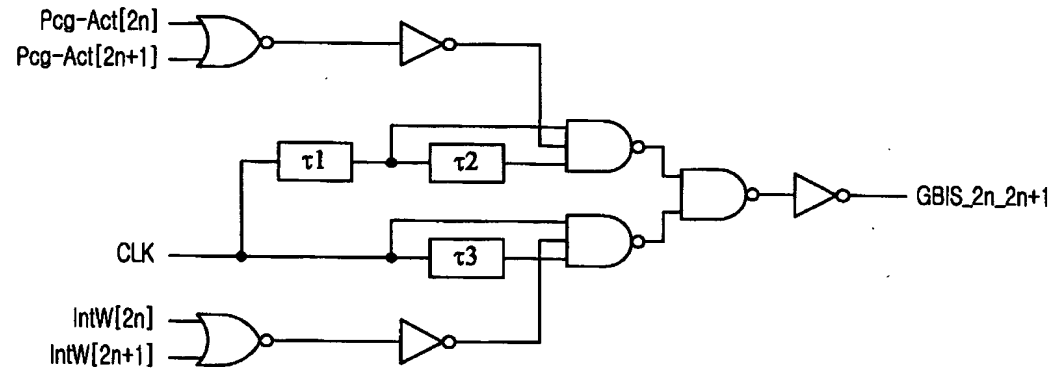
【도 17】


4비트
256
예비블럭
테이블
(EBT)

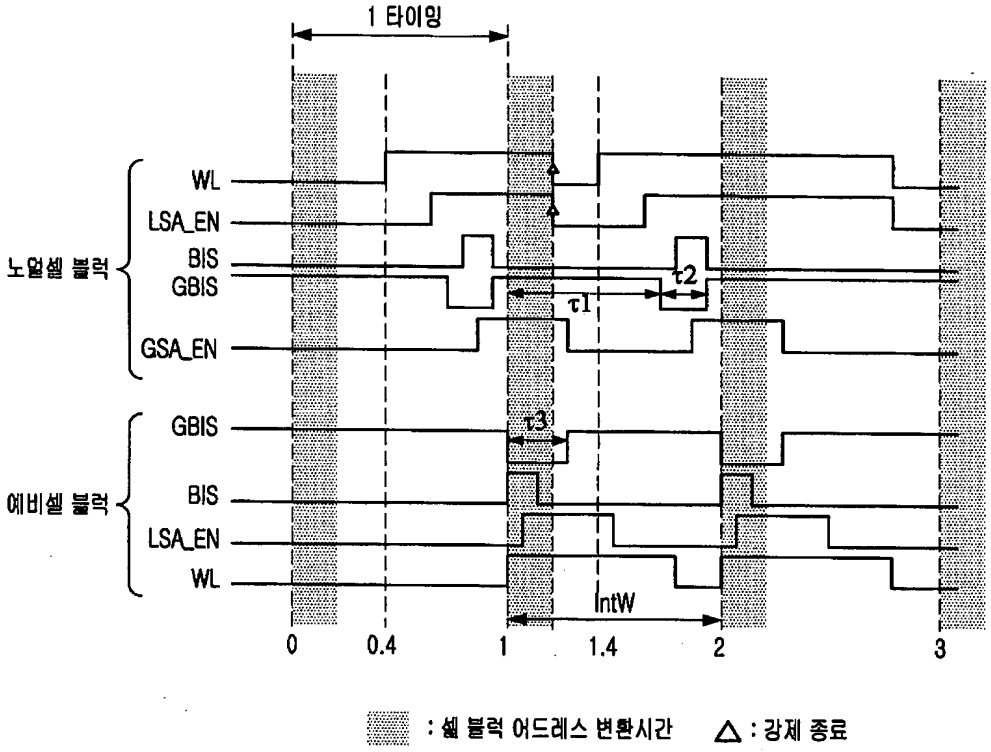
【도 18】


EBT:BL1
A
물리적 어드레스 변환
B
태그데이블에 논리적 어드레스 저장
C
비교
F
EBT 업데이트
G
태그 블럭 동작 구간
블럭 동작 구간
제1 타이밍
EBT:BL0
H
물리적 어드레스 변환
I
태그데이블에 논리적 어드레스 저장
J
비교
D
EBT 업데이트
E
제2 타이밍

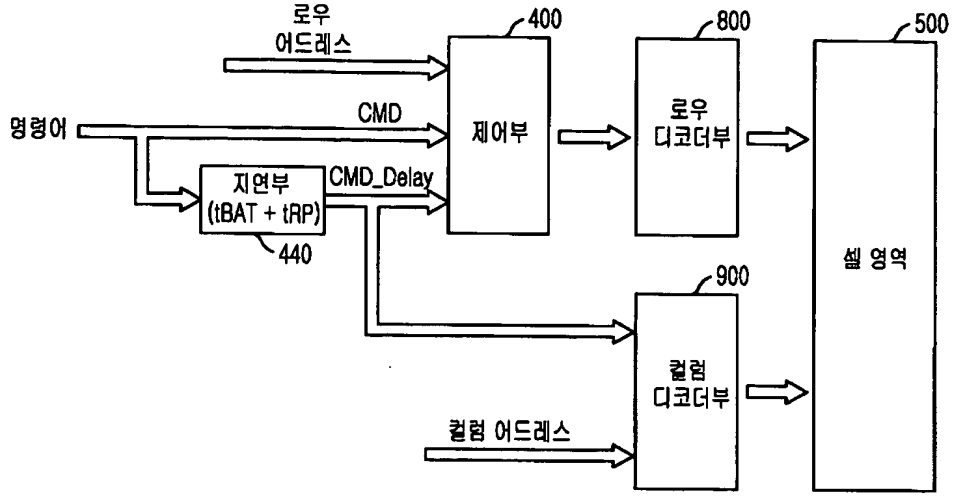
【도 19】


Pcg-Act[2n]
Pcg-Act[2n+1]
τ1
τ2
τ3
CLK
IntW[2n]
IntW[2n+1]
GBIS_2n_2n+1

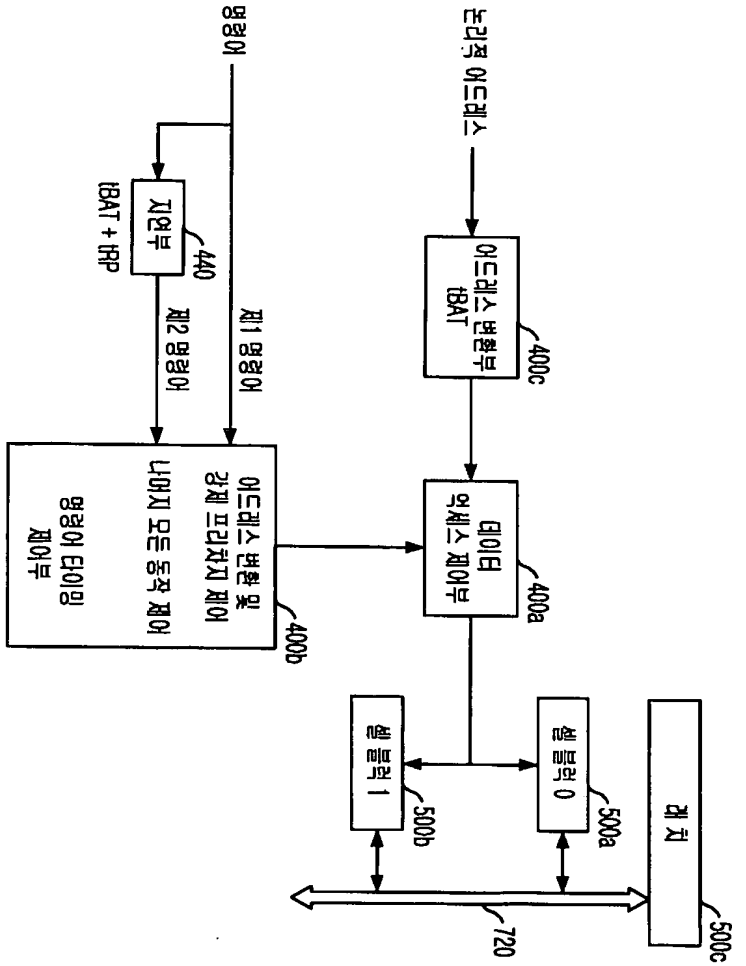
【도 20】


1 타이밍
노멀셀 블럭
WL
LSA_EN
BIS
GBIS
GSA_EN
τ1
τ2
예비셀 블럭
GBIS
τ3
BIS
LSA_EN
WL
IntW
0
0.4
1
1.4
2
3
: 셀 블럭 어드레스 변환시간
△ : 강제 종료

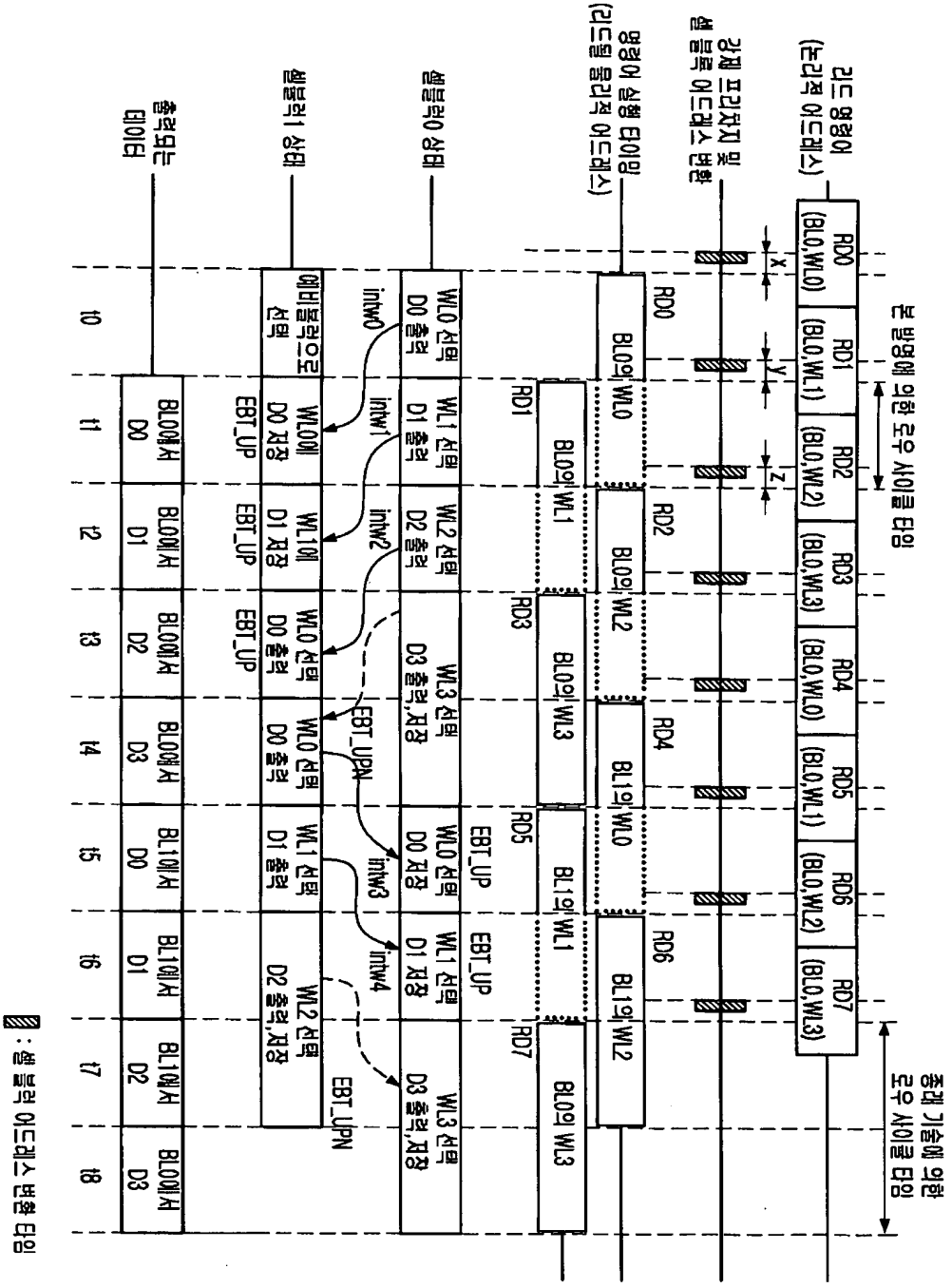
【도 21a】


로우
어드레스
400
800
500
명령어
CMD
제어부
로우
디코더부
지연부
(tBAT + tRP)
CMD_Delay
440
900
셀 영역
컬럼
디코더부
컬럼 어드레스

【도 21b】


명령어
로우 어드레스
tBAT + tRP
지연부
440
제2 명령어
제1 명령어
어드레스 변환부
tBAT
400c
어드레스 변환 및
칩제 프리차지 제어
나머지 모든 동작 제어
명령어 타이밍
제어부
400b
데이터
억세스 제어부
400a
셀 블럭 1
셀 블럭 0
500b
500a
720
레지
500c

【도 22】


리드 명령어 (논리적 어드레스)
RD0 (BL0,WL0)
RD1 (BL0,WL1)
RD2 (BL0,WL2)
RD3 (BL0,WL3)
RD4 (BL0,WL0)
RD5 (BL0,WL1)
RD6 (BL0,WL2)
RD7 (BL0,WL3)
x
y
z
본 발명에 의한 로우 사이클 타임
실제 프리차지 및 셀 블록 어드레스 변환
명령어 실행 타이밍 (리드될 물리적 어드레스)
RD0 BL0의 WL0
RD1 BL0의 WL1
RD2 BL0의 WL2
RD3 BL0의 WL3
RD4 BL1의 WL0
RD5 BL1의 WL1
RD6 BL1의 WL2
RD7 BL0의 WL3
종래 기술에 의한 로우 사이클 타임
셀블럭0 상태
WL0 선택 D0 출력
WL1 선택 D1 출력
WL2 선택 D2 출력
WL3 선택 D3 출력,저장
WL0 선택 D0 저장
WL1 선택 D1 저장
WL2 선택 D2 출력,저장
WL3 선택 D3 출력,저장
셀블럭1 상태
예비블럭으로 선택
WL0에 D0 저장
WL1에 D1 저장
WL0 선택 D0 출력
WL0 선택 D0 출력
WL1 선택 D1 출력
WL2 선택 D2 출력,저장
EBT_UP
EBT_UPN
intw0
intw1
intw2
intw3
intw4
출력되는 데이터
BL0에서 D0
BL0에서 D1
BL0에서 D2
BL0에서 D3
BL1에서 D0
BL1에서 D1
BL1에서 D2
BL1에서 D3
t0
t1
t2
t3
t4
t5
t6
t7
t8
: 셀 블럭 어드레스 변환 타임